UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Fátima Ap. Benthien da Silva Schmitt

Utilização da Programação Funcional no Mundo dos Blocos Geométricos

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de mestre em Ciência da Computação.

Prof. Jorge Muniz Barreto, D. Sc. A.

Florianópolis, Setembro de 2003

Utilização da Programação Funcional no Mundo dos Blocos Geométricos

Fátima Ap. Benthien da Silva Schmitt

Esta Dissertação foi julgada adequada para a obtenção do título de mestre em Ciência da Computação, área de concentração Sistemas de Conhecimento e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

_____

Prof. Fernando Alvaro Ostuni Gauthier, Dr.
Coordenador do Curso

Banca Examinadora

_____

Prof. Jorge Muniz Barreto, D. Sc. A.

_____

Antonio Carlos Zimmermann, Dr.

_____

Profa. Maria Augusta Soares Machado, Dra.

_____

Prof. Mauro Roisenberg, Dr.

"Quando surge um problema, você tem duas alternativas: ou fica se lamentando, ou procura uma solução. Nunca devemos esmorecer diante das dificuldades. Os fracos se intimidam. Os fortes abrem as portas e acendem as luzes."
(Dalai Lama)

Ofereço à Maria, mãe de Jesus, a conclusão deste trabalho.
E também a meus dois Rogerios (esposo e filho) por toda a
compreensão nas minhas ausências da vida familiar.

Agradecimentos

Por imenso prazer e não por obrigação, gostaria de agradecer a todos aqueles que de alguma forma contribuíram para a finalização deste trabalho.

Aos meus familiares, pela ajuda, compreensão e força a mim dispensadas em todas as horas que não pude estar junto a eles, pois estava em companhia dos livros.

A amiga e companheira, Claudia, por me incentivar e estar comigo nessa caminhada.

A UFSC e aos professores, em especial aqueles que mais tive contato e que muito contribuíram para o meu crescimento no conhecimento.

A meu orientador, pelo seu imenso conhecimento e vontade que sejamos seus filhos intelectuais, pois foi em uma de suas aulas que a semente deste trabalho foi germinada.

Aos membros da banca por suas considerações relevantes sobre a minha dissertação.

A Rogério, meu marido, companheiro de todas as horas, sempre ao meu lado, apoiando, estimulando e ajudando no possível para que mais um de meus sonhos se concretizasse.

Aos amigos que fiz nessa caminhada: Eliane, Claudia, Reinaldo, Luciene, Cíntia, Gláucio e outros, que sempre unidos e prontos para ajudarem estavam ao meu lado e também aos amigos, que infelizmente, perdi.

Em especial ao meu filho, Rogério Junior, pelas horas que não pude acompanhá-lo em suas brincadeiras e anseios infantis, mas que esse meu esforço sirva de aprendizado para ele, que saiba que sempre conseguimos alcançar o que almejamos, mesmo com muitas dificuldades, esforço e dedicação.

A amiga Maike, pois acompanhando o seu sofrimento, procurava encontrar forças nas horas difíceis e seguir seu exemplo de luta.

A minha mãe e amiga, pelo seu imenso apoio e desejo de ter uma filha doutora. Estou a caminho querida mãe.

Aos diretores da empresa Benner, por terem me concedido o tempo necessário à realização deste mestrado.

Sei que não se aprende sem tentativas, sem escorregões, sem tropeços, sem ensaio e agradeço a todos referenciados acima que me ajudaram a enxergar o caminho, a levantar e seguir adiante para a realização deste sonho.

Sumário

Lista de Figuras

Siglas Principais

Siglas:

CAI : Instrução auxiliada por computador.

CAPRE : Comissão de Coordenação das Atividades de Processamento Eletrônico.

CENIFOR : Centro de Informática Educativa.

CSN : Conselho de Segurança Nacional.

EDUCOM : Informática na educação.

FLPL : Fortran List Processing Language.

IA : Inteligência Artificial.

FURB : Universidade Regional de Blumenau.

GEACE : Grupo Executivo de Aplicação de Computadores Eletrônicos.

IAS : Inteligência Artificial Simbólica.

IBGE : Instituto Brasileiro de Geografia e Estatística.

LEC : Laboratório de Estudos Cognitivos.

LISP : List Processing.

MEC : Ministério da Educação e Cultura.

MIT : Massachusetts Institute of Technology.

OEA : Organização dos Estados Americanos.

OOP : Programação Orientada a Objetos.

PF : Programação Funcional.

PROLOG : Programmation en Logique.

SBC : Sociedade Brasileira de Computação.

SEI : Secretaria Especial de Informática.

SEED : Secretaria de Educação à distância.

UFBA : Universidade Federal da Bahia.

UFMG : Universidade Federal de Minas Gerais.

UFPE : Universidade Federal de Pernambuco.

UFRGS : Universidade Federal do Rio Grande do Sul.

UFSC : Universidade Federal de Santa Catarina.

UFRJ : Universidade Federal do Rio de Janeiro.

UNICAMP : Universidade de Campinas.

KRC : Kent Recursive Calculator.

Publicações

1. Fátima Aparecida Benthien da Silva Schmitt, Jorge Muniz Barreto e Paulo Sérgio da Silva Borges. Uma Proposta Teórica da Distribuição de Conhecimento no Atendimento. I Encontro de Ciência e Tecnologia - ECTec 2002 - Lages - SC - Brasil.

2. Fátima Aparecida Benthien da Silva Schmitt e Jorge Muniz Barreto. Uso da Programação Funcional de Forma Lúdica. XVIII Congresso Regional de Iniciação Científica e Tecnológica em Engenharia - CRICTE 2003 e IV Feira de Protótipos - Universidade do Vale do Itajaí - Itajaí - SC.

Resumo

O objetivo deste trabalho é mostrar uma modalidade de uso do elemento tecnológico, computadores na educação, como uma ferramenta cognitiva onde o computador passa de uma máquina de “ensinar” para uma máquina de “pensar com”, sendo um elo da mente humana, em um processo de elaboração e exploração do conhecimento, articulando-o com o processo de desenvolvimento do pensamento, para o entendimento da essência filosófica do Paradigma Funcional e seu impacto sobre o uso dos computadores no ensino.

Pretende-se fornecer aos alunos de uma maneira lúdica, através de algumas regras de um jogo de construção de figuras geométricas de duas dimensões, a oportunidade de formalizar seus pensamentos de uma forma natural, sem a necessidade de, antes de conseguir formalizar seu pensamento ter que se preocupar em aprender uma linguagem para se comunicar com o computador.

Desenvolveu-se um protótipo de um ambiente onde o aluno poderá fazer testes, analisar e resolver problemas, utilizando os conhecimentos já adquiridos na matemática, podendo dessa forma se concentrar na solução do problema cometendo erros e corrigindo-os. Proporcionando um tipo de aprendizado por exploração e descoberta, ou seja, o aluno tem plena liberdade de guiar seu próprio aprendizado, podendo realizar os experimentos que considerar mais interessante, facilitando sua compreensão da estratégia utilizada para a resolução do problema, e o entendimento de como utilizar o paradigma funcional para atingir seus objetivos.

Favorecendo desta forma, uma introdução menos traumática do aluno iniciante e inexperiente a um ambiente de programação através do paradigma funcional e mostrando uma nova maneira de pensar sobre o ato de construir o conhecimento.

Palavras-chave: Computadores na educação, Paradigma Funcional, Ensino.

Abstract

The objective of this work is to show a modality of use of the technological element, computers in the education, as a cognitive tool, the computer passes of a machine of "teaching "for a machine of "thinking with "being a link of the human mind, in an elaboration process and exploration of the knowledge, articulating it with the process of development of the thought, for the understanding of the essence philosophical of the Functional Paradigm and its impact on the use of the computers in the teaching.

It intends to supply the students in an entertaining way through some rules of a game of construction of geometric illustrations of two dimensions, the opportunity to formalize its thoughts in a natural way without the need of before getting to formalize its thought to have to worry in learning a language to communicate with the computer.

A prototype of an atmosphere was developed where the student can make tests, to analyze and to solve problems, already using the knowledge acquired in the mathematics, being able to not in that way to concentrate in the solution of the problem making mistakes and correcting them. Providing a learning type for exploration and discovery, that is to say, the student has full freedom of guiding its own learning, could accomplish the experiments that to consider more interesting, facilitating its understanding of the strategy used for the resolution of the problem, and the understanding of as to use the functional paradigm to reach its objectives.

Favoring this way, a less traumatic introduction of the student beginner and inexperienced to a programming atmosphere through the functional paradigm and showing a new way to think on the act of building the knowledge.

Keywords: Computers in the education, Functional Paradigm, Teaching.

Capítulo 1

Introdução

“Se escuto, esqueço. Se vejo, recordo. Se faço, compreendo”. (provérbio chinês)

1.1 Motivação

Diante do desafio de ensinar, da melhor maneira possível, programação de computadores aos alunos iniciantes vê-se nesse trabalho a possibilidade de um estudo através dos caminhos que melhor levem a compreensão do aluno, de uma maneira mais dinâmica e facilitando o entendimento de como se “comunicar” com o computador, através da PF (Programação Funcional), que é uma linguagem natural e que aplica os conhecimentos do aluno já adquiridos nas aulas de matemática.

A programação de computadores freqüentemente é vista como uma atividade difícil, acessível apenas a algumas pessoas. Há vários fatores envolvidos no ensino da programação responsáveis por essa crença, entre eles: o tipo de linguagem que permite a interação entre o usuário e o computador, as linguagens de programação possuem uma sintaxe extremamente difícil e são implementadas em inglês, obrigando os alunos a dedicarem muito esforço no aprendizado da linguagem de programação enquanto código lingüístico artificial. A aprendizagem do código da linguagem ocorre de forma descontextualizada, isto é, o aprendizado lógico, abstrato em si não tem nenhuma relação com a atividade de programação. A programação passa a ser uma atividade de resolução de problemas bastante complexa e a linguagem serve somente para codificar a resolução final a fim de ser passada para o computador. O pressuposto por detrás dessa afirmação é o de que o aluno

deve aprender a linguagem computacional por que a utiliza naturalmente e não por que decora os comandos e as regras de uso dos mesmos. Pode-se pensar que isso é análogo ao que ocorre com a criança quando ela está adquirindo a sua língua natural. Nessa fase, a criança gradativamente vai aprendendo a sua língua não por que ela "decora" as palavras ou recebe "aulas" de como falar, mas, por que ela conversa com as pessoas e, durante essa interação vai construindo a sua linguagem. A aquisição da linguagem pela criança é um processo longo, complexo, no qual o interlocutor representa um papel de fundamental importância. Da mesma maneira, o sujeito que aprende uma linguagem de programação, precisa interagir com o computador, que neste caso é o seu interlocutor, a fim de construir um conhecimento sobre o funcionamento desse código artificial.

A motivação para este trabalho é o reconhecimento da importância de ensinar da maneira mais natural possível. E o desafio de implementar solução de problemas com "brincadeiras ", motivando o aprendizado com a utilização do computador, desenvolvimento do senso crítico e da criatividade, levando o aluno a despertar sua inteligência e capacidade de raciocínio naturalmente.

O interesse nessa maneira natural de aprendizagem iniciou-se em 1997, no curso de pós-graduação em nível de especialização em Tecnologias de Desenvolvimento de Sistemas realizado na FURB. Em 1998 a autora, desta dissertação, Fátima [52], através de algumas pesquisas, deparou-se com as dificuldades encontradas pelos alunos de 2^0 grau e faculdades brasileiras com a introdução do pensamento formal para solução de problemas. Então, sentiu-se na época, a necessidade de um auxílio baseado no computador como meio facilitador da aprendizagem dos conceitos básicos de algoritmos, para auxiliar os alunos, através da visualização e execução no computador dos algoritmos e o acompanhamento passo a passo dessa execução. Criando dessa maneira uma interação entre a disciplina e o computador, fazendo disso um processo natural e agradável de aprender, então foi desenvolvido um protótipo para Ensino de Algoritmos, sob a orientação da professora Clarisse Odebrecht, como parte integrante da monografia da pós-graduação citada acima.

Com o início do mestrado em Ciência da Computação na UFSC e o contato com as técnicas de IA (Inteligência Artificial) e PF novamente o interesse pelo ensino de forma natural e lúdica veio à tona.

Para Papert [43], não se aproxima do conhecimento juntando peças de conhecimento e colocando-as umas sobre as outras, mas a aquisição se dá por processo de descoberta, de significação e de trocas de experiências com o meio, onde o aluno empregando seus próprios conhecimentos já adquiridos, sua forma de ver o mundo, vai estabelecendo conexões e construindo novos relacionamentos entre os conhecimentos adquiridos, ou

mesmo construindo novos conhecimentos de maneira intuitiva e natural, sem o formalismo tradicional adotado nos sistemas de ensino.

O trabalho tem sua maior ênfase nas atividades que visam promover a aprendizagem ativa, onde o aluno, sujeito do processo, é incitado a pensar sobre o problema de movimentar as peças do jogo, a explicitar a solução escolhida da forma que considerar mais adequado, segundo seu próprio estilo de pensamento. Testando e depurando suas idéias tanto sobre o dispositivo montado quanto sobre o programa que o comanda, observando e descobrindo os recursos funcionais que estão sendo utilizados para a realização do problema apresentado.

Desenvolveu-se um trabalho em torno de soluções, não se limitando a determinados conteúdos e nem a temas específicos. Ao mesmo tempo em que alimenta a racionalidade, o ambiente proposto incita o exercício da dúvida na tentativa de compreender as ações e representações do aluno. Revelando a sua identidade, abolindo a polarização objetividade-subjetividade, favorecendo a interação entre diferentes formas de produção do conhecimento.

Para Silva [54], em se tratando de aspectos cognitivos, podemos considerar importante a visão construcionista do conhecimento, onde processos centrais do indivíduo, como organização do conhecimento, processamento de informações, atitudes quanto à tomada de decisões, solução de problemas através da criação de estratégias, entre outros, são amplamente valorizados, sendo assim, a ferramenta computacional deve ser suficientemente ampla para permitir a múltipla contextualização e o aprendizado através dos mais diversos meios de assimilação.

1.2 Objetivos

1.2.1 Objetivo geral

O entendimento da essência filosófica do paradigma funcional e seu impacto sobre o uso de computadores no ensino, com a utilização do protótipo desenvolvido, de um jogo para crianças de 7 ou 8 anos.

Propõe-se a apresentação de soluções para os problemas a serem resolvidos, mais próximas ao raciocínio humano, ou seja, a utilização da memória humana, sem se preocupar com a memória da máquina.

Caracterizando no protótipo desenvolvido, que na PF, não se raciocina com endereço de memória da máquina, como na programação imperativa e sim com funções (dados de entrada e resultados).

Utilizar-se do ambiente desenvolvido: “Mundo dos Blocos Geométricos” para uma aprendizagem dinâmica, que enfatiza a construção do conhecimento, facilitando a descrição, a reflexão e a depuração de idéias, com o objetivo de apresentar a interface funcional, permitindo ao aluno, num primeiro momento, interagir com um paradigma de programação que possibilitará uma reflexão crítica sobre as formas de pensar, abrindo “caminhos” para a utilização de outras ferramentas, com o auxílio do computador.

1.2.2 Objetivos específicos

- Relatar o uso da informática na educação, para através da aprendizagem lúdica, alcançar-se um nível de abstração maior do uso do computador como meio de ensino;
- Descrever a facilidade da utilização da PF, evolução e utilização da mesma;
- Referenciar um dos trabalhos mais conhecidos, desenvolvido com a utilização da PF, mostrando que apesar de ainda ser pouco utilizada é muito mais simples que a programação imperativa e muito eficiente principalmente para alunos iniciantes;
- Desenvolvimento do protótipo de um jogo de construção de figuras geométricas, possibilitando ao aluno comandar o computador funcionalmente, abstraindo os conceitos de PF e raciocínio geométrico de duas dimensões.

A implicação deste trabalho na sociedade é a necessidade de preparar os alunos para tornarem-se legítimos cidadãos que constroem seu próprio conhecimento. Em vista da globalização onde, cada vez mais, as relações humanas, visando à capacidade de tomada de decisões conjuntas, são exigidas.

1.3 Organização deste documento

O primeiro capítulo contém a descrição das motivações e interesses do trabalho.

No segundo capítulo descreve-se a utilização da informática na educação, seus benefícios e anseios.

No terceiro capítulo faz-se uma descrição sobre a essência filosófica da PF e seus benefícios ao serem aplicados no ensino de crianças e alunos iniciantes.

No quarto capítulo descreve-se um resumo de uma das mais antigas linguagens funcionais e que teve muitos dialetos: LISP e a utilização do LOGO, que foi desenvolvido em LISP.

No quinto capítulo descreve-se que para fazer o trabalho foi escolhido ensinar algo diferente, ou seja, ensinar de uma forma diferente da tradicional, saindo das aulas expositivas e desmotivantes para um ambiente de aprendizado com entretenimento. Aqui se procurou preencher uma outra lacuna, através da PF demonstrar de uma forma lúdica aos alunos a capacidade de programar o computador da forma mais natural possível, desenvolvendo o raciocínio do aluno. Para isso iniciou-se fazendo uma análise de quais figuras geométricas seriam necessárias para a construção do protótipo, onde o aluno desenvolverá seu raciocínio geométrico, espacial e lógico.

No sexto capítulo são apresentadas as considerações finais sobre o trabalho desenvolvido e sugestões para trabalhos futuros.

Capítulo 2

Informática na educação

“O que sabemos é uma gota; o que ignoramos é um oceano!” (Isaac Newton)

Denomina-se de computador uma máquina de processar dados, numéricos ou simbólicos, que funciona através da execução de programas. Ao contrário das inúmeras máquinas que conhecemos, tais como: de lavar roupa, liquidificador, enceradeira, aspirador de pó, e tantas outras, que realizam uma única função, o computador é uma máquina multiuso. Pode-se usá-lo como uma máquina de escrever sofisticada, como uma máquina de fax, como uma prancheta de desenho, como um fichário eletrônico, como uma planilha de cálculos e de tantas outras formas. É exatamente como o nosso conhecido vídeogame, para mudar de jogo basta trocar o programa. No vídeogame, cada novo jogo é determinado por um novo programa. Em linhas gerais pode-se entender um computador como uma máquina capaz de:

- interpretar dados que lhe são fornecidos, produzindo resultados em forma de novos dados ou comportamentos, usando para isso conceitos que lhe foram antecipadamente informados e,

- aceitar a descrição de novos conceitos e considerá-los na interpretação de novas situações.

Um dos méritos do computador no campo da educação é o de tentar resolver um dos grandes problemas da educação: como respeitar o ritmo da aprendizagem, como evitar defasagem entre os tempos propostos pela escola e o tempo necessário ao aluno numa atividade particular em um determinado momento da vida? A escola tende a queimar

etapas e não poucas vezes o que se quer dos alunos é por demais superior as suas possibilidades.

Fagundes [15] ressalta que a escola tem se constituído em um lugar onde grupos diversificados de “especialistas” tentam transmitir conjuntos estanques de informações aos alunos que, se espera, aprendam as mesmas coisas num mesmo tempo. Os sistemas de avaliação empregados comparam as respostas dos alunos a padrões universais previamente definidos. E os conteúdos ensinados são escolhidos e hierarquizados, ou pelos professores, ou pelos órgãos governamentais, independentes das condições estruturais e funcionais daqueles que devem aprender, com o objetivo de proporcionar uma base de fundamentos comuns para qualquer profissão, visando a formação do cidadão.

Os objetivos da escola ao introduzir a informática, devem ser centrados em cada aluno, ou seja, entender e desenvolver o perfil cognitivo de cada aluno, avaliando as suas capacidades individuais, trabalhando as múltiplas inteligências, adequando os alunos às maneiras particulares de aprender, e entendendo que nem todos os alunos aprendem tudo o que há para ser aprendido.

Segundo Tjara [56], a inteligência não pode ser medida, ela não é um produto acabado, pois dependendo do contexto sócio-econômico-cultural, uma ação pode ser valorizada em um ambiente e em outro ambiente não ter nenhuma significância. Gardner apresenta sete competências intelectuais autônomas do ser humano:

- Inteligência lingüística: habilidade ou capacidade em lidar com os desafios relacionados com a linguagem.
- Inteligência lógico-matemática: habilidade de resolução de problemas por meio da dedução e da observação.
- Inteligência corporal-cinestésica: habilidade em utilizar movimentos corporais para superar desafios de uma determinada realidade.
- Inteligência musical: habilidade de produzir e perceber as notações musicais.
- Inteligência espacial: habilidade em abstrair interação com o ambiente, o espaço e o ciberespaço para elaborar um produto ou resolver um problema.
- Inteligências intrapessoais: habilidade em conhecer os aspectos internos de uma pessoa.
- Inteligência interpessoal: habilidade em perceber as intenções e desejos dos seus interlocutores e, com isso, resolver ou minimizar problemas de comunicação e relacionamento.

Baseados nessas características individuais de cada aluno, é que o computador na educação deve ser utilizado como meio facilitador do desenvolvimento das inteligências e apoio da construção do conhecimento.

Dependendo da visão educacional e da visão pedagógica, a informática na educação assume diversos significados.

A visão instrucionista defende o paradigma da continuidade da informação para o aluno, com a continuidade da prática pedagógica vigente, auxiliada com o uso da informática, já a visão construcionista, reforça a construção do conhecimento individualizado através de ambientes que utilizem o computador como fonte de motivação.

Com a transformação constante que a sociedade moderna está passando, principalmente no conceito de globalização, cada vez mais o sucesso da nação depende de seu cidadão, através de sua capacitação, sua cultura, sua educação. Para que os alunos sejam sujeitos ativos no processo de aprendizagem, tendo capacidade não só de explorar o meio, mas também de gerar conhecimento, desenvolver seu potencial cognitivo e afetivo, através do autoconhecimento e auto-enriquecimento, o processo de ensino, através do uso da informática com a utilização de ricos ambientes de aprendizagem é fundamental.

Os computadores estão cada vez mais importantes em nossa sociedade. Sua presença cultural aumentada a cada mês e sua chegada nas escolas permite que as crianças adquiram experiência, conhecimento e entusiasmo sobre o que estes conhecimentos podem fazer [23].

Mas esse processo de ensino precisa ser questionado: - Como está funcionando? A que serve? Para que serve? Por quê?

E o próprio computador deve ser investigado: - Como pode ser usado? Por quê? Que efeitos terão seus diferentes usos?

Poderá esse instrumento tão importante na revolução informática constituir-se num auxílio para o desenvolvimento do “homem social”?

Servirá o computador para ajudar a transformação do “ensino”, entrando no sistema educacional para alimentar o processo de aprendizagem “natural” e “espontâneo” das crianças e adolescentes?

Em vez de desconhecer a história da aprendizagem da criança e do jovem que nela ingressa, a escola precisaria investigar essa história para lhe dar continuidade, pois a criança aprende desde que nasce. Aprende quando ainda não esta submetida a qualquer ensino formal. A criança pequenina é naturalmente o mais espontâneo e persistente aprendiz. Quando tem saúde e liberdade de movimentos, mantém-se em constante atividade: explora o ambiente, faz experiências, formula teorias sobre como são e como funcionam os objetos, testa hipóteses. Está sempre criando idéias [8].

O importante nesse processo de aprendizagem espontânea é que suas teorias podem ser falsas, as relações que descobre podem ser inadequadas, as idéias fantasiosas, porque está organizando e reorganizando tudo continuamente, retestando, construindo e reconstruindo.

Defende Abreu [1], em sua tese, que é preciso que ocorra uma mudança de paradigma em relação ao próprio processo de constituição do saber, ou seja, que os fatos sejam encarados com outro marco epistêmico no qual os esforços pedagógicos estejam centralizados na dinâmica do pensamento, considerando as condições de produção de saber do aluno, a natureza de seus instrumentos cognitivos e seu processo de funcionamento e desenvolvimento.

Em Almeida [3] a autora considera importante ensinar com o computador e não somente desenvolver ferramentas puramente teóricas. Assim, foi para a sala de aula, usando como recurso pedagógico às implementações de ambientes computacionais de ensino desenvolvidos seguindo a modelagem proposta em sua tese, verificando erros e acertos, observando as reações dos alunos, suas preferências, ouvindo opiniões, críticas e sugestões no dia a dia. Ou seja, vivenciando, na prática, o que denominam de “mitos e verdades da Informática na Educação”.

2.1 No Brasil

Para Moraes [39], no final da década de 1950, começaram a chegar no Brasil os primeiros computadores. Eles vieram, entre outros, para o governo de São Paulo, para o Jóquei Clube de São Paulo e o Instituto Brasileiro de Geografia e Estatística (IBGE). No meio acadêmico, foi a Pontifícia Universidade Católica do Rio de Janeiro a pioneira na utilização desse equipamento. Em seguida, a Universidade de São Paulo e o Instituto Tecnológico da Aeronáutica também criaram os seus próprios computadores. Muitos desses equipamentos foram trazidos ao País com os incentivos do Grupo Executivo de Aplicação de Computadores Eletrônicos (GEACE), criado em 1959.

Relata Almeida [2] que o uso de computadores na área de educação vem sendo utilizados desde os anos 60 quando aconteceu à primeira experiência educacional, na área de física na Universidade Federal do Rio de Janeiro, outros registros, relatam que a história da informática na Educação teve seu surgimento no meio acadêmico em 1971, voltado inicialmente para o ensino de física na Universidade de São Carlos, em São Paulo. Ainda nessa década foram feitas experiências com crianças que tinham dificuldades de aprendizado e também o desenvolvimento da linguagem Logo.

Outros registros garantem que o computador ensaiou seus primeiros passos em 1980 com a criação da Comissão Especial n^0 01: Informática na Educação : CE-IE criada pela Secretaria Especial de Informática - SEI, e logo após quando ocorreram os primeiros seminários de Informática na Educação (Brasília -1981 e Bahia-1982) que motivaram o desenvolvimento de projetos em universidades.

Em 3 de outubro de 1979, foi criada a SEI (Secretaria Especial de Informática), como órgão complementar do CSN (Conselho de Segurança Nacional), assumindo as funções acumuladas pela CAPRE (Comissão de Coordenação das Atividades de Processamento Eletrônico).

Em março de 1980, a SEI cria a Comissão Especial de Educação, um segmento de apoio ao MEC, para assumir o papel de gerador de normas e diretrizes no novo e amplo campo que se abria para a educação.

No início de 1983, os membros da Secretaria reuniram-se para elaboração do projeto EDUCOM - Informática na Educação - (MEC, CNPQ, Finep e a SEI) que tinha como objetivo investir em grupos de pesquisa interessados em criar recursos humanos dentro das universidades Federais.

Em fins de 1984, o CENIFOR (Centro de Informática Educativa), assume a responsabilidade pela coordenação e supervisão técnica da execução do projeto EDUCOM.

Segundo Franco [20], o projeto EDUCOM, caracterizou-se como um experimento de natureza intersetorial de caráter essencialmente educacional, onde cada entidade pública federal participava, não apenas custeando parte dos recursos estimados, mas também acompanhando seu planejamento, sua execução e avaliação, de acordo com sua vocação institucional, conjugando esforços para garantia de maior impacto dos objetivos pretendidos, assim em cada EDUCOM havia um projeto autônomo.

Cada uma das universidades envolvidas com o Projeto EDUCOM, desenvolveu seus próprios sub-projetos. A lista das instituições participantes era a seguinte:

- UFMG - Universidade Federal de Minas Gerais.
- UFPE - Universidade Federal de Pernambuco.
- UFRGS - Universidade Federal do Rio Grande do Sul.
- UFRJ - Universidade Federal do Rio de Janeiro.
- UNICAMP - Universidade Estadual de Campinas.

O projeto LOGO na UNICAMP iniciou-se a partir de um estágio, em 1973/74, da Profa. Afira V. Ripper no Laboratório LOGO do MIT (Instituto de Tecnologia de Mas-

sachusetts), onde teve a oportunidade de conhecer o trabalho dos Profs. Seymour Papert e Marvin Minsky (ver figura 2.1). Estes professores foram convidados a visitar a UNICAMP em 1975, e, como resultado desta visita, formou-se um grupo interdisciplinar de pesquisa, que contava, à época, com os Profs. Fernando Curado, do Departamento de Computação, Maria Fausta Campos e Cláudia Lemos, do Departamento de Lingüística, Raymond Paul Shepard e Márcia de Brito, do Departamento de Psicologia Educacional. A profa. Afira V. Ripper foi contratada pela Faculdade de Educação, Departamento de Psicologia Educacional, especificamente para atuar neste projeto.

Figura 2.1: Marvin Minsky, um dos grandes apoiadores de Papert

Em 1978, a UFRGS desenvolveu uma pesquisa sobre o sistema CAI(Instrução Auxiliada por Computador).

Em 1980, os pesquisadores do LEC (Laboratório de Estudos Cognitivos), no Rio Grande do Sul, também entraram em contato com Papert no MIT, e começaram com uma experiência com quatro crianças, já em 1981. Pode-se ver na figura 2.2, um dos projetos do LEC.

Figura 2.2: Novos mundos desde 1980.

O local que possui mais informações acumuladas na área da informática na Educação com a linguagem LOGO é o LEC, que vem desenvolvendo pesquisas desde 1981. Contudo, os seus relatórios revelam [16][17][18], ainda muitas questões sem respostas em relação aos comportamentos cognitivos possíveis, quanto ao uso de computadores com a linguagem LOGO.

O Projeto LOGO da UNICAMP e a criação do Laboratório de Estudos Cognitivos (LEC) da UFRGS foram, indiscutivelmente, os que mais influenciaram o uso generalizado e quase exclusivo da Linguagem LOGO, na maioria das escolas brasileiras que passaram a adotar os computadores, por quase uma década.

No fim do ano de 1989, o projeto EDUCOM foi desativado por falta de verbas, e pela expectativa de mudança de governo.

Logo depois surgiu o projeto Formar que visava à capacitação de professores de 1^0 e 2^0 graus e à implantação de estruturas de suporte dentro das secretarias estaduais de educação. Foram criados, então, os Centros de Informática Aplicada a Educação de 1^0 e 2^0 graus (Cied), os Centros de Informática na Educação Tecnológica (Ciet) e os Centros de Informática na Educação Superior (Cies). As instituições deviam, a partir daí, desenvolver individualmente suas propostas de ensino.

A implantação de Cieds em vários estados fez com que a informática educativa chegasse a alunos e professores de 1^0 e 2^0 graus. Foi estabelecida uma parceria entre o MEC e o governo mexicano através da Organização dos Estados Americanos (OEA). Juntos, Brasil e México formularam o Proninfe, que tratava da implantação da informática educativa fundamentada em projetos pedagógicos sólidos e atualizados, com aplicação na rede pública de ensino de 1^0 ao 3^0 grau.

A Secretaria de Educação a Distância, SEED/MEC, em novembro de 1996 lança o Programa Nacional de Informática na Educação, Proinfo, com os seguintes objetivos:

- Melhorar a qualidade do processo de ensino-aprendizagem;

- Propiciar uma educação voltada para o desenvolvimento científico e tecnológico e;
- Educar para uma cidadania global, numa sociedade tecnologicamente desenvolvida.

Em 1997 foi implantado o Projeto Ensino On Line que é um projeto do governo estadual paulista, e é estruturado na distribuição de computadores e softwares educacionais nas escolas de ensino fundamental e médio e na formação de professores-multiplicadores.

O ponto central desse projeto é a formação dos professores.

Podemos perceber que a história da informática na Educação não é tão recente, mas se compararmos o Brasil com países como EUA e França, veremos que o uso do computador na educação no Brasil é recente.

2.2 Visão construcionista do uso do computador na educação

O autor mais importante nesta abordagem é Seymor Papert (ver figura 2.3) , que completou seus estudos com Jean Piaget, e que após investigações nas áreas de matemática, IA e psicologia, construiu uma linguagem computacional, com o apoio de Marvin Minsky, integrando os conhecimentos acerca da construção de conhecimentos, numa perspectiva interacionista-construtivista.

Figura 2.3: Seymor Papert, um dos autores mais importantes da visão construcionista

O computador é apresentado como uma poderosa ferramenta cognitiva, auxiliar na construção do conhecimento, capaz de desenvolver o raciocínio lógico-dedutivo, a organização do pensamento, facilitar a expressão da criatividade e a postura crítica.

Através de ferramentas mentais ou cognitivas, o aluno dentro de um contexto real constrói seu conhecimento de uma forma intelectual (crítico, criativo, e pensamento de

alta ordem) e social (cooperação). Estas ferramentas, mentais ou cognitivas são consideradas parceiras intelectuais porque melhoram a capacidade de pensar do aluno ao contrário de ferramentas produtivas que só melhoram a produção do aluno. As ferramentas cognitivas que podem ser usadas no processo de ensino/aprendizagem via computador são: planilhas, banco de dados, correio eletrônico, rede, fórum de discussão, programação de computador, hipermídia, hipertexto, e ambientes de aprendizagem envolvendo várias ferramentas.

A utilização do computador, como um instrumento de observação do funcionamento cognitivo e emocional do sujeito, traz ao professor uma nova visão do aluno, que assume uma postura mais crítica em relação ao processo como um todo, revelando-se capaz de elaborar métodos próprios nos caminhos de sua aprendizagem, exercitando sua criatividade e desenvolvendo um progressivo e seguro domínio da máquina.

O computador na sala de aula pode ser uma ferramenta cognitiva para o aluno, criando-se um ambiente de aprendizagem, onde os alunos possam desenvolver habilidades em um contexto que faça parte da sua vida real, que haja aprendizagem colaborativa, ativa, facilitada e os alunos possam construir a sua interpretação do mundo real, interiorizando os conhecimentos e organizando-os.

Baseando-se na visão construcionista, como uma nova maneira da representação do conhecimento é a linha que será seguida nesse trabalho.

Denominou-se de construcionista a abordagem pela qual o aprendiz constrói, por intermédio do computador, o seu próprio conhecimento. Papert [44] usou esse termo para mostrar um outro nível de construção do conhecimento: a construção do conhecimento que acontece quando o aluno constrói um objeto de seu interesse, como uma obra de arte, um relato de experiência ou um programa de computador. Na noção de construcionismo de Papert existem duas idéias que contribuem para que esse tipo de construção do conhecimento seja diferente do construtivismo de Piaget. Primeiro, o aprendiz constrói alguma coisa, ou seja, é o aprendizado por meio do fazer, do “colocar a mão na massa”. Segundo, o fato do aprendiz estar construindo algo do seu interesse e para o qual ele está bastante motivado. O envolvimento afetivo torna a aprendizagem mais significativa.

O computador é uma máquina que fascina, amedronta, apaixona [8].

Em resumo, um computador é um conjunto de circuitos eletrônicos, capaz de tratar e/ou memorizar uma informação.

O uso da informática dentro da educação possibilita a construção de novos conceitos, usando programas, com jogos criativos, promovendo o desenvolvimento das potencialidades do aluno. O computador promove uma nova visão do mundo ao educando, modificando suas representações mentais, tornando-se uma ferramenta muito poderosa em um

ambiente de trabalho cooperativo, dinamizado pelo professor [41].

As escolas treinam e desenvolvem muito mais as faculdades sentantes que as faculdades pensantes das crianças. Treinam e formam sentistas, de tanto sentar e ouvir, sem agir [10]. O que podemos esperar de quem passou toda sua vida escolar SENTADO, tanto física quanto intelectualmente, a ouvir como as coisas devem ser "sabidas"?

Observa-se que o uso da informática na educação é mais que uma necessidade é um salto qualitativo, ampliando parte da capacidade mental do aluno de processar informações. Utilizando o computador não como um meio de transferir a informação, mas sim como uma ferramenta na qual o aluno, pode formalizar seus conhecimentos analiticamente e construir seus conceitos espontâneos, permitindo dessa maneira ao aluno expandir sua capacidade natural de pensar lógico-formalmente, aumentando, assim, a possibilidade de codificar, processar e decodificar informações.

2.3 Como usar as máquinas para ensinar

Chaves [13] defende a tese de que toda criança deveria aprender a programar, porque esse aprendizado, além de útil por si mesmo, traz embutida a aprendizagem de uma série de conceitos, habilidades e atitudes que são importantes, ele diria até essenciais, para o desenvolvimento intelectual e cognitivo. Uma certa atmosfera de mistério e até mesmo de magia cerca o primeiro contato de alguém com o computador. Embora sabido que se trata apenas de uma máquina com circuitos, teclas, etc, há algo no computador que o faz parecer não só quase vivo, mas também inteligente.

A primeira coisa que o aprendizado de programação demonstra é que o computador só faz aquilo que é ensinado a fazer. Sem um programa e, portanto, sem programador, o computador é inútil, entretanto, tem uma excelente memória e uma capacidade servil de executar ordens com precisão e rapidez. No processo de aprender, o aluno aprende quem dá ordens a quem, quem é que ensina e instrui, quem é que está no controle. Ajudando o aluno a desenvolver a autoconfiança, advinda do fato de que ele é capaz de fazer uma máquina poderosa e até misteriosa obedecer às suas ordens. É de vital importância que os alunos se convençam que são eles que devem, mesmo porque podem, controlar as máquinas e não vice-versa.

A programação permite a realização do ciclo descrição-execução-reflexão-depuração-descrição. O programa representa a idéia do aprendiz e existe uma correspondência direta entre cada comando e o comportamento do computador. As características disponíveis no processo de programação ajudam o aprendiz a encontrar seus erros, e ao professor

compreender o processo pelo qual o aprendiz construiu conceitos e estratégias envolvidas no programa [57].

Se perguntarmos a duas crianças:

O que vocês aprenderam na escola hoje?
Resposta da primeira criança: Eu aprendi que 2 + 3 = 5.
Resposta da segunda criança: Eu aprendi como fazer uma adição.

Conclui-se pela primeira resposta que a criança recebeu um ensino do tipo saber, que privilegia sua memória. Em compensação, a segunda resposta põe em evidencia um ensino do tipo saber-fazer, que necessita de suas faculdades de análise (tratamento).

As vantagens do uso do computador para o ensino do tipo saber-fazer são:

- A máquina transmite um saber ou uma habilidade (ensina).
- A máquina é um meio de ligação que permite que os indivíduos se comuniquem, através da interposição da tecnologia.
- A máquina favorece (serve de pretexto para) a comunicação direta, do mesmo modo que o gesto e a palavra.
- O aluno progride segundo seu próprio ritmo.
- O aluno é autônomo.
- O aluno conseguirá assimilar melhor o programa.

Aprender por analogia, de fato, constitui uma das formas mais naturais e mais promissoras de aprendizagem humana. As pessoas desenvolvem estruturas cognitivas estendendo, analogicamente, estruturas anteriores conhecidas, as pessoas tentam aprender coisas novas fazendo uso de suas aprendizagens passadas. Novos conceitos são tipicamente pensados em termos de conceitos anteriores, assim como situações anteriormente experimentadas costumam ser tomada como uma guia para novas situações, pelo menos numa fase inicial [33].

Estruturas mentais são codificações internas criadas na memória a partir de estímulos externos. Conhecer a natureza destas estruturas constitui o interesse fundamental tanto da Ciência Cognitiva (e da Engenharia Cognitiva) quanto da IA, em particular. Por exemplo,

são de interesse questões como: Qual a representação interna das estruturas cognitivas? O que distingue uma estrutura predicativa (que se concentra em rede de relações e subestruturas) de uma estrutura funcional (onde predomina o pensamento em termos de efeitos e de seqüências de ações)? Como se relacionam essas construções estruturais no processo de aprendizagem do agente cognoscente?

Cognição é o ato mental ou processo através do qual o conhecimento é adquirido, incluindo também a percepção, a intuição, e o raciocínio.

O desafio em mudar a forma de ensinar e aprender no contexto da escola reside em criar ambientes de aprendizagem que incentivem o uso de diferentes ferramentas de comunicação para enriquecer a exploração e a investigação de um problema para dar origem a outros problemas. Além disso, esses ambientes devem instigar o estudante a observar, questionar, discutir, interpretar, solucionar, analisar e esses são alguns dos exemplos de competências, segundo o que nos coloca Perrenoud [45].

Meirieu [37] valoriza a pedagogia das situações-problema como uma prática que desafia os alunos a buscar respostas cuja construção resulta necessariamente em uma nova aprendizagem. Dar a chance ao aluno de participar na elaboração de seu conhecimento é um dos pontos fundamentais da concepção de aprendizagem e esta participação deve ser orientada tendo em vista os conceitos a serem construídos, bem como as tarefas a serem realizadas para que esta construção se efetive.

A IAS (Inteligência Artificial Simbólica) vai conquistando avanços conceituais fundamentais na compreensão da inteligência, na compreensão dos contextos ou significados, na compreensão de como representar conhecimentos, de como as pessoas relembram as coisas e, sobretudo, na compreensão de como as pessoas aprendem e pensam.

Aprendizagem é construção e modificação de representações daquilo que está sendo experimentado.

O computador em si é neutro, depende de como iremos usá-lo.

Na visão de Gasperetti [22], caso o professor empregue uma técnica instrucionista, com o uso do computador, esquecendo a participação dos estudantes, o professor arrisca a limitar sensivelmente (ou bloquear por completo) o aprendizado. Há um provérbio chinês que diz: “Se escuto, esqueço. Se vejo, recordo. Se faço, compreendo”.

2.4 Classificação de tipos de usos

No ensino, o computador poderá atuar, nas seguintes modalidades:

1. Aprendizagem sobre computadores - que é a aprendizagem sobre o funcionamento

do computador.

2. Aprendizagem através dos computadores - onde o computador será usado como um recurso instrucional, tanto para dar uma aula completa, na forma de instrução programada, como para dar suporte ao professor em experimentos, simulações, exercícios, como um instrumento de apoio ou áudio-visual.

3. Aprendizagem com computadores - é a maneira de usar o computador de modo interativo aluno-máquina, em que o ato de elaborar e pensar o programa pertence ao aluno, em uma visão dinâmica e cognitiva da aprendizagem. Este modo significaria uma superação do uso do computador como mero recurso didático, dando-lhe o papel de uma ferramenta cognitiva.

O uso do computador pode ser catalogado também em função da palavra informática, distinguindo a aprendizagem “pela” e a aprendizagem “da” informática.

“Pela” prática da informática, o aluno adquire métodos e técnicas que pode transpor para outras disciplinas, especialmente a matemática. “Pela” utilização de um sistema informático, o aluno adquire ou verifica modelos de pensamento, num contexto de resolução de problemas, ou de comunicação, engendrada pela presença de um sistema informático motivante [8].

O computador é o melhor instrumento para se praticar a aprendizagem “da” informática, seja por intermédio de um objeto-programa construído e fornecido ao aluno, seja por uma programação efetiva.

O profissional que trabalha com informática, com um grande “I” é um mito. O conhecimento de uma linguagem de programação leva certos profissionais a se considerarem muito a sério. A palavra informática se refere tanto ao material (hardware) quanto à coleção de programas (software), cada vez menos dissociados, desde a introdução dos microprocessadores no mercado.

A atividade de programação é enriquecedora; contudo, ela representa apenas um componente das atividades possíveis com um computador.

Utilizando o computador, como ferramenta produtiva, o aluno adquire conhecimentos ou habilidades pré-estabelecidas que o auxiliam a realizar uma tarefa, tornando-o mais produtivo. Como os softwares de exercício e prática, muito usado nos anos 70 e 80 e ainda hoje, onde os alunos resolvem os problemas, entrando com as respostas e já tem uma realimentação da acuracidade da sua resposta, promovendo uma automaticidade. Ainda hoje este tipo de programa é usado para aprender línguas, memorizar informações, cursos nas empresas, aprender computação e outros.

Outros tipos de programas que se moldam a esse enfoque são os programas tutoriais que tem como finalidade responder às diferenças individuais na aprendizagem, fornecendo instrução de reforço, quando os alunos respondem incorretamente, limitando o aluno a adquirir um conhecimento pronto, funcionam como uma espécie de guia. Desenvolvidos nos anos 80 e 90 por pesquisadores de IA, os chamados Sistemas Tutoriais Inteligentes, que vieram acrescentar ao tutorial, modelos especialistas que descrevem o raciocínio ou estratégias que um especialista usaria para resolver um problema.

Os jogos de simulação, utilizando o computador, como ferramenta mental, são considerados os melhores, favorecem:

- autoconfiança, companheirismo, capacidade de comunicação, mudanças na relação professor-aluno, segurança;
- estruturação do conhecimento;
- e auto-estima pela possibilidade de alcançar o sucesso.

O aluno aprende a brincar com o seu próprio erro, de tal forma que, errando, pode criar novas situações e testar novas hipóteses. Assim, o erro deixa de ser um “fantasma”, pois além de ser visualizado simbolicamente, passa a ser elemento de comparação diacrítica, favorecendo a construção do próprio caminho da descoberta.

Na verdade, esta classificação é interessante apenas para uma exposição didática. Na prática, nenhum destes enfoques de produtos de software educacional funciona isoladamente. Atualmente, os sistemas procuram ser híbridos e o que se precisa buscar cada vez mais, é navegar por todos eles o tanto quanto possível. Pode-se classificá-los em grupos dessa maneira:

- Os programas aplicativos ou abertos, do ponto de vista da informática (editores de texto, planilhas eletrônicas, bancos de dados, programas gráficos). Oferecem a possibilidade de construção de macro-comandos similares aos procedimentos de linguagem de programação clássica. Sendo programas especializados, destinam-se a atividades com conteúdos preciso como esquematizar, classificar objetos ou resolução de problemas numéricos, conforme o objetivo a ser atingido.
- Os micro-mundos são sistemas informatizados onde o aluno deve explorar um domínio com um mínimo de ajuda do sistema, combinando as primitivas de uma linguagem de programação. O aluno aprende a aprender utilizando o ambiente para espelhar seus conhecimentos e construir novos objetos.

- Os coursewares são produtos de software educacional clássicos que, a partir de uma situação interativa entre o aluno e um problema, leva o aluno a resolvê-la. A gama de atividades possíveis é vasta, mas cada seqüência é fechada por respostas interpretáveis pelo programa. São considerados como parte de um ambiente que favorece pouco a iniciativa do aluno e são muito especializados em relação aos objetivos pedagógicos. A concepção deste ambiente repousa no diálogo interativo e a aprendizagem consiste no aluno realizar a seqüência de procedimentos associados a determinados conceitos.

- Os tutores inteligentes são como alguns coursewares onde a característica de resolução de problemas, acrescentou-se o componente tutorial, onde são representados o modelo do aluno, o conhecimento e a técnica do professor e a especialização do conhecimento a ser ensinado. A idéia é permitir aprendizagem de alto nível (a lógica e compreensão) através da tutoria entre o sistema do professor e o sistema do aluno. A concepção destes sistemas é análoga às ajudas "on-line" disponíveis, por exemplo, nos aplicativos.

- O Hipertexto é comumente definido como uma forma não linear de armazenamento e recuperação de informações. Isto significa que a informação pode ser examinada em qualquer ordem, através da seleção de tópicos de interesse. Desta forma, um hipertexto tem como principal característica à capacidade de interligar pedaços de textos ou outros tipos de informação entre si através do uso de palavras-chave. Hipertextos, hipermídia e multimídia podem ser adequados à educação. A interação ativa de um indivíduo com a aquisição do saber é pedagogicamente interessante. Com a multimídia interativa, isto é, com a possibilidade de uma dimensão reticular, não linear, há o favorecimento de uma postura exploratória diante do conteúdo a ser assimilado. Desta forma, hipermídia estaria relacionada a uma aprendizagem ativa.

- Os jogos são programas de entretenimento e apresentam grande interatividade e recursos de programação muito sofisticados.

Os ambientes inteligentes de aprendizagem permitem ultrapassar a oposição simplista entre os defensores da aprendizagem por indução (atividades exploratórias do sujeito) e os defensores dos tutoriais. Estes sistemas devem ser capazes de favorecer a aquisição de conceitos e procedimentos associados a um domínio do conhecimento, permitindo ao aluno transformar suas experiências em conhecimento organizado. Nestes ambientes, considera-se que a melhor forma de aprender é conceber ferramentas que assistam aos alunos para que possam comunicar-se de forma eficaz.

2.5 Uma maneira lúdica de aprendizado

O par jogo e computador são dois artifícios muito úteis que devem ser utilizados para a motivação da aprendizagem e que contribuem muito para tal, sendo vistos como o par perfeito, porém, segundo Chaiben [12], nenhuma outra recente inovação na educação tem gerado tantos questionamentos quanto a dos computadores em sala de aula. Quando utilizados com racionalidade podem desempenhar um papel valioso no processo educacional, estimulando o interesse do aluno, resolvendo problemas, ou realizando simulações. Por outro lado, quando utilizados sem muitos critérios podem não só produzir efeitos indesejáveis como também consumir recursos expressivos, aplicando-se também essas preocupações à utilização dos jogos na educação.

2.5.1 Aprendizagem por ensaio e erro

Descreve Barros [6] que foi o pesquisador americano do comportamento Edward Lee Thorndike(1874-1949) que utilizou gatos pela primeira vez na famosa “caixa-problema”.

Foi com base nessas experiências que Thorndike desenvolveu o método de solução de problemas conhecido como ensaio ou tentativa e erro.

Para realizar esse experimento, Thorndike aprisionou um gato faminto numa caixa de madeira. Do lado de fora, junto à caixa, havia um pedaço de peixe ou carne. Para sair da caixa e alcançar o alimento, o gato teria de abrir o trinco da porta. Essa situação, geralmente, provocava imediata atividade por parte do gato, que tentava passar através das barras da caixa, enfiava a pata nas aberturas, enfim, buscava um jeito de escapar, demonstrando o comportamento chamado de “ensaio e erro”.

No esforço para escapar, o gato, acidentalmente, abria o trinco e conseguia chegar ao alimento. Thorndike permitia que o gato comesse apenas um pedacinho do alimento e, imediatamente, recolocava-o na caixa. Geralmente, na segunda tentativa e em várias outras subseqüentes, o gato repetia os movimentos desordenados e impulsivos. Mas, à medida que repetia a experiência, a atividade do animal tornava-se cada vez mais restrita, e gradualmente eram eliminados os movimentos inúteis. Finalmente, o gato, ao ser colocado outra vez na caixa, dirigia-se quase imediatamente à porta, acionava o trinco e saía. Havia aprendido.

A partir disso, Thorndike chegou à conclusão: *todo processo de solução de problemas consiste em uma série de tentativas casuais, uma das quais conduz eventualmente ao resultado desejado.*

A criatividade também é um processo de solução de problemas, embora de um tipo

todo especial.

Com o método de ensaio e erro, consegue-se às vezes descobrir uma coisa nova, porém isto somente por meio de uma série bastante demorada de experiências. A aprendizagem segundo o método de ensaio e erro é um aprendizado através do êxito.

2.5.2 Aprender por fazer

Às vezes, os espíritos inovadores, ainda que solitários e infortunados, podem mudar o mundo e atravessar os tempos. Nos anos 30, um professorzinho francês de saúde frágil resolveu mudar a escola. Fez isso de modo provocativo, levando para a sala de aula uma máquina: uma impressora, tipos móveis de chumbo, tinta e objetos de ofício de uso comum numa tipografia. As crianças se transformavam em tipógrafos, imprimindo o jornal, e depois em jornalistas. Saíam da classe com o professor Célestin e iam entrevistar o prefeito, o padre, o chefe de polícia. Estudavam ciências procurando as flores mais raras, e depois escreviam e imprimiam em seus relatos. Dezenas, centenas de documentos tornavam-se jornais e depois eram distribuídos em outras escolas. Tornou-se tão famoso o método Freinet, que logo outras escolas adotaram a tipografia.

Freinet compreendera a importância do criar, do fazer [22].

2.5.3 Aprendizagem por descoberta

Em Barros [6] é relatado que Bruner não inventou o método de aprendizagem por descoberta, pois ele já tinha sido proposto por filósofos e educadores anteriores.

Bruner, nos dias de hoje, é o divulgador do método e lhe deu fundamentação teórica, e por seus estudos e observações, propõe que os professores adotem esse método.

Na abordagem expositiva, o professor já traz o conteúdo pronto e o aluno limita-se passivamente a escutá-lo. Na abordagem hipotética, o professor traz o assunto sob forma de problema ou questão a ser resolvida, e ajuda o aluno a resolvê-lo, discutindo com ele as alternativas apresentadas.

As atividades lúdicas têm o poder sobre a criança de facilitar tanto o progresso de sua personalidade integral, como o progresso de cada uma de suas funções psicológicas intelectuais e morais.

O lúdico é um traço da personalidade que persiste da infância até a juventude e a idade adulta, com função muito importante no estilo cognitivo dos indivíduos, ou seja, na alegria, no senso de humor e na espontaneidade [29].

Se o lúdico é tão discutido por psicólogos e pensadores, não seria este o momento de a

escola parar e refletir também sobre a importância do lúdico (jogos e brinquedos)? Quais os benefícios? Como utilizar essas atividades lúdicas para a aquisição do conhecimento como um todo?

Como benefício didático, as brincadeiras transformam conteúdos maçantes em atividades interessantes, revelando certas facilidades através da aplicação do lúdico. Outra questão importante é a disciplinar, pois quando há interesse pelo que está sendo ensinado, a criança canaliza suas energias para aquilo que esta sendo apresentado e faz com que automaticamente a disciplina aconteça.

Em Gasperetti [22], é relatado um estudo sobre a importância da brincadeira: *Muitos etólogos estudaram a importância da brincadeira para os primatas. Anos a fio, observaram o comportamento dos leões e dos gorilas e perceberam que os filhotes incapazes ou com alguma impossibilidade de brincar permaneciam imaturos e, quando adultos, não eram capazes de enfrentar as adversidades da natureza, estando destinados a morrer cedo.*

De fato, nos primatas, a brincadeira é uma espécie de treinamento, de teste geral para enfrentar a realidade. O pequeno leão brinca com a cauda da mãe. Depois, a leoa oferece ao filhote uma pequena presa moribunda. Finalmente, leva-o à caça. O pequeno aprende, tranqüilo, sem stress, e amanhã será capaz de enfrentar a vida real, de caçar e de evitar o perigo.

O homem não se afasta muito desse modelo de aprendizado natural. Segundo alguns estudos efetuados nos Estados Unidos nos anos 60, as crianças impossibilitadas de brincar são menos inteligentes, têm menos senso prático e sentem aversão a seus próprios semelhantes.

Foi Gouvea [24] quem usou a frase: antes das descobertas das numerosas vitaminas, já haviam encontrado a mais importante para as crianças, a vitamina “R” (recreação); para ela a recreação ou atividade lúdica é tudo o quanto diverte e entretém o ser humano e envolve a ativa participação .

2.6 Vantagens do uso do computador para ensinar

Kreutz [27], salienta que o uso do computador no ensino possibilita:

- Ao aluno dirigir o seu próprio aprendizado;

- Uma verificação rápida das respostas do aluno, devido a sua capacidade de cálculo;

- Uma apresentação interessante de conteúdo, aliando texto, gráficos, som e animação;
- Uma comunicação interativa, isto é, recebe respostas do aluno, verifica-as e envia novas questões ou correções das respostas, gerando assim uma progressão pedagógica;
- Superar obstáculos geográficos quando conectado a uma rede de comunicações, permitindo ao aluno aprender em qualquer momento ou lugar;
- Ao aluno repetir incessantemente as mesmas lições de acordo com seu interesse;
- Uma formação individualizada por aluno, ou seja, adaptada ao seu ritmo e interesse próprios;
- Ao professor concentrar-se mais nos aspectos criativos da formação, deixando os aspectos repetitivos para o computador;
- Registro das respostas do aluno e o acesso seletivo as informações registradas, devido a sua capacidade de memorização.

Pagano [42], resume que o espectro de utilizações do computador no ensino pode ser classificado:

- Como agente modificador do ambiente de vida do aluno;
- Como jogo educativo;
- Como aparelho de laboratório;
- Como enciclopédia;
- Como interlocutor pedagógico.

No artigo de Alonso [4] é enfatizado que o ambiente informatizado de aprendizagem privilegia a interação entre sujeitos em um clima colaborativo e extremamente desafiador, os quais constroem seus conhecimentos a partir das suas vivências e de acordo com seus diferentes ritmos de aprendizagem, mediados pelo professor.

De maneira geral, a principal vantagem na adoção de programas de educação no processo de ensino/aprendizagem, é a possibilidade da individualização do processo de aprendizagem, respeitando o ritmo de cada aluno garantindo a mesma qualidade a todos os alunos que tiverem acesso ao sistema.

Capítulo 3

Programação Funcional

“A mente que se abre a uma nova idéia, jamais voltará ao seu tamanho original.” (Albert Einstein)

3.1 Introdução

O que caracteriza a PF é que não se raciocina com endereço de memória da máquina, como na programação imperativa. E sim com funções, que para solucionar um problema, através de um conjunto de dados de entrada, vão produzir um conjunto de dados de saída, encontrando a solução para o problema apresentado.

Neste capítulo, descreve-se a evolução das linguagens, a diferença entre programação imperativa e funcional e as facilidades do uso da PF.

3.2 Histórico

Cálculo Lambda pode ser considerada a primeira linguagem de PF, embora nunca tenha sido projetada para ser realmente executada em um computador. É um modelo de computação projetado por Alonzo Church nos anos 30 que oferece um modo muito formal de descrever um cálculo de uma função.

A primeira linguagem de PF criada para computadores foi o LISP “List Processing”, desenvolvida por John McCarthy no MIT no fim dos anos 50. Mesmo não sendo uma linguagem de programação puramente funcional, o LISP introduziu a maioria das carac-

terísticas hoje encontradas nas modernas linguagens de PF.

Existem diversas linguagens funcionais: Lisp, Hugs, Clean, Haskel, Miranda, Variantes de Lisp, KRC (um acrônimo para Kent Recursive Calculator).

As linguagens funcionais mais conhecidas são o LISP e o Prolog (Programação em Lógica). A linguagem Scheme também é freqüentemente citada, por ser uma variante simplificada do LISP. Diversas outras linguagens funcionais são encontradas na literatura, por exemplo, AspecT, Caml, Clean, Erlang, FP, Gofer, Haskell , Hope, Hugs, Id, IFP, J, Miranda, ML, NESL, OPAL e Sisal.

As principais linguagens funcionais são:

- A linguagem Haskell foi lançada no fim dos anos 80 em uma tentativa de juntar muitas idéias na pesquisa de PF. É similar a ML em razão de usar uma sintaxe semelhante; tem escopo estático, é fortemente tipificada e usa o mesmo método de inferência de tipos. Ela difere da ML em termos de ser puramente funcional; não tem variáveis, nenhuma instrução de atribuição e não inclui nenhum recurso imperativo de qualquer tipo. Isso a separa de quase todas as outras linguagens de programação.

- A linguagem Scheme, um dialeto do LISP, surgiu no MIT em meados da década de 70. Ela é caracterizada pelo pequeno tamanho, pelo uso exclusivo que faz do escopo estático e pelo tratamento que dá às funções como entidades de primeira classe. As funções Scheme podem ser os valores de expressões e elementos de listas, e podem ser atribuídas a variáveis e passadas como parâmetros. As primeiras versões do Lisp não forneciam todas estas capacidades.

  Como uma linguagem pequena com sintaxe e semântica simples, a Scheme adapta-se bem a aplicações educacionais, como, por exemplo, a cursos de PF e também a introduções gerais a programação.

  Exemplo:

  (let ((a 3)(b 4) (let ((double (* 2 a)) (triple (* 3 b))) (+ double triple))))

  is 18.

- A ML foi criada pela Universidade de Edimburgo, é uma linguagem de PF de escopo estático como a Scheme. Usa uma sintaxe semelhante à do Pascal. A ML tem declarações de tipo e usa inferência de tipos (o que significa que variáveis não precisam ser declaradas) e é fortemente tipificada. O tipo de cada variável e de

expressão pode ser determinados sem tipo. A ML tem manipulação de exceções e uma facilidade modular para implementar tipos de dados abstratos.

Na ML, nomes podem ser vinculados a valores,tendo as instruções de declaração de valor a forma:

val novo_nome = Expressão

Por exemplo:

val distancia = tempo + velocidade

A instrução val vincula um nome a um valor, mas o nome não pode ser revinculado a um novo valor mais tarde. Bem, em certo sentido, pode. Realmente, se for revinculado um nome com uma Segunda instrução val, isso causará uma nova entrada no ambiente não relacionada à versão anterior do nome. De fato, seu tipo não precisa ser o mesmo. Elas simplesmente adicionam um nome ao ambiente atual e circulam a um valor, como forma especial LET do LISP. O uso normal da val é uma expressão let, cuja forma geral é Let val novo_nome = Expressão_1 in expressão$_2$.

- MacLisp melhorou na versão Lisp 1.5 a noção de variáveis especiais e o manejo de erros. MacLisp introduziu também o conceito de funções que poderiam receber um número variável de argumentos, macros, arrays, saídas não locais dinâmicas, aritmética rápida, o primeiro bom compilador de Lisp e enfatizou a velocidade de execução. No fim dos anos 70, MacLisp era usado em mais de 50 cidades.

- Interlisp introduziu muitas idéias no ambiente e metodologia da programação em Lisp. Uma das idéias do Interlisp que influenciou o Common Lisp foi uma "*iteration construct*" implementada por Warren Teitelman que inspirou o "loop macro" usado tanto no Lisp Machines e no MacLisp e agora no Common Lisp.

- Prolog é uma linguagem declarativa, não-proprietária, baseada em inferência dedutiva sobre cláusulas de Horn, um subconjunto restrito e muito simplificado do cálculo de predicados de primeira ordem que conserva, entretanto, toda a sua expressividade original. O primeiro interpretador Prolog foi escrito pela equipe do Prof. Alain Colmerauer, na Universidade de Aix-Marseille em 1973. No ano seguinte Robert Kowalski, então no Imperial College, em Londres, formalizou a semântica procedimental do Prolog.

Histórico:

- Nos anos 70: R.A. Kowaski fornece as bases teóricas do Prolog. Alain Comerauer e colegas fazem a primeira implementação do Prolog.
- Nos anos 80: David Warren da Universidade de Edinburgh constrói o primeiro compilador. Torna-se popular com o Turbo-Prolog da Borland. Os japoneses escolhem o Prolog como linguagem básica para o projeto da 5^a geração.

Características:

- Prolog é uma linguagem declarativa, isto é, o usuário "declara" o modelo do seu problema e deixa o Prolog buscar a solução.
- Permite construir protótipos rapidamente.
- Elegância e facilidade de compreensão.
- Linguagem de 5^a geração.
- Modularidade.
- Polimorfismo.
- Alocação e desalocação automática de memória dinâmica.
- Compilação incremental.
- Meta-programação (programas que manipulam outros programas).

Exemplo:

Dada as relações existentes numa família através da linguagem Prolog. O fato de Carlos ser pai de José pode ser expresso em Prolog como:

pai (carlos, jose).

Seja o functor pai como o nome dessa relação; os nomes carlos e josé são os argumentos dessa relação.

A árvore genealógica de uma família pode ser definida pelo seguinte programa:

pai (carlos, jose).
mae (maria, jose).
pai (davi, ana).
mae(paula, ana).
pai (davi, pedro).

mae (paula, pedro).
pai (jose, marcos).
mae (ana, marcos).
pai (vitor, carol).
mae (bruna, carol).
pai (pedro, joao).
mae (carol, joao).
pai (pedro, carla).
mae(carol, carla).

O programa é composto de cláusulas. Cada uma dessas cláusulas declara um fato sobre a relação pai ou mae.

Prolog pode apresentar algumas perguntas ou consultas sobre a relação pai ou mãe, quando o programa é comunicado ao sistema Prolog.

3.2.1 Evolução das linguagens

Conforme Hasemer [25] os computadores são baseados na arquitetura Von Neumman, que foi o primeiro matemático que desenvolveu um ciclo de processamento para o computador digital, o qual é a base de vários ciclos utilizados até hoje, embora geralmente esse conceito seja atribuído a outro matemático Alan Turing, é natural que as primeiras linguagens de programação tivessem uma significativa proximidade com essa arquitetura, e foi exatamente o que aconteceu. Ainda sem libertar-se da arquitetura Von Neumman, mas voltando-se mais na direção do programador do que da máquina, iniciou-se o projeto que resultou na linguagem Fortran (um dos marcos na Ciência da Computação), por John Backus. Mesmo considerada como uma linguagem de alto nível, Fortran tinha a característica de que programar reflete a codificação de um procedimento (conjunto ordenado de passos) que deve ser executado pela máquina. E que pode ser vista até hoje em várias linguagens de programação como C, Pascal, C++, Delphi, Cobol, Clipper, etc.

Os primeiros computadores digitais, que surgiram na década de 40, eram usados e, de fato, foram inventados para aplicações científicas. Tipicamente, as aplicações científicas têm estruturas de dados simples, mas exigem um grande número de computações aritméticas com ponto-flutuante. As estruturas de dados mais comuns são os arrays e as matrizes, as estruturas de controle mais comuns são os laços de contagem e de seleções. As linguagens de programação de alto nível, inventadas para aplicações científicas, foram

projetadas para suprir essas necessidades [53]. A concorrente delas foi à linguagem assembler: desse modo, a eficiência era a primeira preocupação. A primeira linguagem para aplicações científicas foi o FORTRAN. O ALGOL 60 e a maioria de suas descendentes também se destinam a serem usadas nessa área, ainda que tenham sido projetadas também para outras áreas relacionadas.

O uso de computadores para aplicações comerciais iniciou-se na década de 50. Equipamentos especiais foram desenvolvidos para tal propósito, juntamente com linguagens especiais. A primeira linguagem de alto nível bem sucedida para negócios foi o COBOL (ANSI, 1985), que apareceu em 1960.

A primeira linguagem de programação desenvolvida para aplicações de IA amplamente utilizada foi a funcional LISP, que surgiu em 1959. No início da década de 70, surgiu uma abordagem alternativa para tais aplicações - a programação lógica, usando a linguagem PROLOG.

A programação orientada a objetos nasceu com a linguagem Simula. O suporte para programação orientada a objeto, agora, faz parte das linguagens de programação mais populares, inclusive a ADA 95 (AARM, 1995) , o Java e o C++.

Um melhor acompanhamento da evolução das linguagens pode ser visto na figura 3.1.

3.3 Programação Imperativa x Funcional

A PF pode ser contrastada com a programação imperativa. Na PF parecem faltar diversas construções freqüentemente (embora incorretamente) consideradas essenciais em linguagens imperativas, como C ou Pascal.

Em uma programação estritamente funcional, não há alocação explícita de memória, nem declaração explícita de variáveis. No entanto, essas operações podem ocorrer automaticamente quando a função é invocada; a alocação de memória ocorre para criar espaço para os parâmetros e para o valor de retorno, e a declaração ocorre para copiar os parâmetros dentro deste espaço recém-alocado e para copiar o valor de retorno de volta para dentro da função chamadora. Ambas as operações podem ocorrer nos pontos de entrada e na saída da função. Isso assegura que o resultado da função será o mesmo para um dado conjunto de parâmetros não importando onde, ou quando, seja avaliada.

Transparência referencial facilita muito ambas as tarefas de comprovar a correção do programa e automaticamente identificar computações independentes para execução paralela.

Laços, outra construção de programação imperativa, está presente através da construção funcional mais geral de recursão. Funções recursivas invocam-se a si mesmas, permitindo que uma operação seja realizada várias vezes. Na verdade, isso prova que laços são equivalentes a um tipo especial de recursão chamado recursão reversa. Recursão em PF pode assumir várias formas e é em geral uma técnica mais poderosa que o uso de laços. Por essa razão, quase todas as linguagens imperativas também a suportam (sendo FORTRAN 77 e COBOL exceções notáveis).

O meio procedural pretende imitar a máquina Von Neumann, que recebeu esse nome em homenagem a Von Neumann (figura 3.2), onde o computador é entendido como uma máquina que obedece ordens e o programa como uma prescrição de solução para o problema.

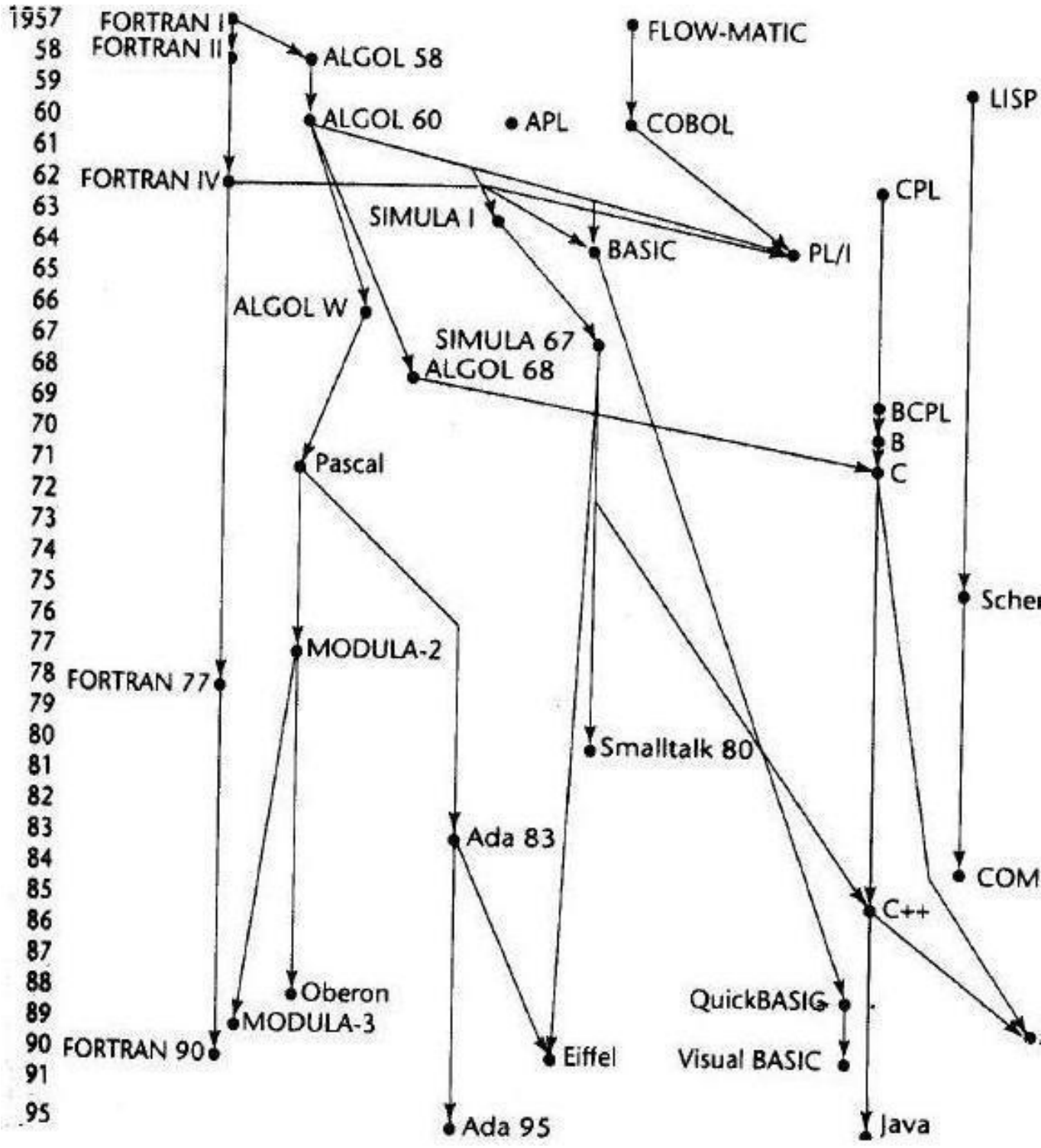

Figura 3.1: Evolução das linguagens de programação

Figura 3.2: Foto de Von Neumann, criador da máquina Von Neumann

Pelo uso de uma seqüência de comandos, pretende-se modificar o estado da máquina. Entende-se aqui, o estado da máquina como um conjunto de valores que definem a atual situação ou comportamento de um sistema e comando, uma ordem (ou imposição) transmitida ao computador pelo programa. Há vários comandos em uma linguagem do paradigma de programação procedimental, tais como, atribuição, alternativo, condicional, etc. Um programa, portanto, é uma série de instruções sobre como modificar o estado da máquina.

Como as diferenças descritas acima, há uma “essencial” que pode ser melhor representada no exemplo demonstrado por Lins [30]:

Considere a equação matemática x = 1. Esta equação tem uma solução, nominalmente x = 1.

Agora considere a equação x = x + 1. Esta equação não tem solução, ou seja, não existe nenhum valor de x igual a seu sucessor, supondo aqui que x é um objeto do “tipo” inteiro. Ainda mais, se considerarmos um mesmo contexto contendo as duas equações (x = 1 e x = x + 1), uma tentativa de resolver o sistema de equações resultante levará a um absurdo, ou seja, 1 = 2.

Agora considere o uso da mesma notação numa linguagem de programação como FORTRAN. Seus “comandos” são “ordens” que levam à modificação do conteúdo da

memória, onde estão armazenados os valores correspondentes aos nomes (variáveis) que fazem parte do programa. Estas ordens podem ser "controladas" por uma família de outros comandos iterativos e seletores. Desta forma, com x representando uma posição de memória qualquer, de tal forma que o "sistema de equações":

x = 1

x = x + 1

É resolvido com x tendo o valor 2. Por causa do caráter seqüencial das linguagens imperativas, a simples inversão da ordem das equações acima resultará em x tendo o valor 1 ao fim da "execução" do "programa".

Apesar da maioria dos programadores ter se acostumado com este modelo de computação, e de muitos nunca o terem questionado, há de se convir que ele não é natural, no que diz respeito à matemática que conhecemos e aprendemos desde a escola primária. Esta diferença gigantesca entre a matemática e as linguagens de programação imperativas, e a conseqüente incapacidade das teorias matemáticas de lidar com os problemas computacionais apresentados por tais linguagens, levou a um movimento que tende a aproximar a programação da matemática.

A notação matemática tem uma semântica estática, isto é, os nomes usados são definidos uma só vez, e seu valor depende somente do contexto da equação que os define [36]. Um dos maiores problemas de ensinar programação imperativa para principiantes é convencê-los de que o modelo computacional correspondente à notação pseudomatemática que a maioria das linguagens de programação usa não tem a menor conexão com o que normalmente se entende por matemática.

As linguagens de programação funcionais não são matemática, apenas mais próximas dela do que Fortran, Pascal, C ou Ada.

A principal característica do meio funcional é imitar o comportamento de funções; o computador atua como uma máquina que avalia funções e o programa consiste da definição de funções e composição de funções. O computador assume o papel de uma máquina funcional. Pode-se ter funções recursivas, obtendo-se algo comparável à estrutura de repetição contida em linguagens procedimentais e as funções são utilizadas de forma muito mais flexível, não tendo que obedecer uma seqüência de comandos, pois uma função não afeta a outra. Pelo uso mais flexível de funções, se ganha concisão e elegância, melhor parametrização e modularidade do programa e formas convenientes de representarem dados infinitos.

As linguagens funcionais são um passo a frente no processo evolutivo das linguagens de programação. Com essas linguagens o programador abstrai-se da máquina e diz apenas o que fazer ao invés de dizer o que e como fazer [30].

Pode-se definir:

- Programação imperativa: a computação baseia-se na existência de um “estado” que vai evoluindo ao longo do tempo. O estado é modificado usando um “comando de atribuição” e existe um “operador de sequenciação”. (Linguagens: Basic, Pascal, C).

- Programação funcional: a computação consiste na avaliação de “expressões” nas quais podem ocorrer chamadas de funções. O objetivo da avaliação de uma expressão é a produção de um “resultado” ou “resposta”. Não há “atribuição” nem variáveis cujo valor possa ser alterado. Cada função limita-se a produzir um resultado a partir dos seus argumentos. (Linguagens: Lisp, ML, Miranda).

No exemplo abaixo da definição do fatorial em definição matemática, PF e em programação imperativa, pode-se comparar às facilidades e benefícios do uso da PF, em relação à programação imperativa.

- **Definição matemática**

```
fatorial X = 1, se X = 0 ,


fatorial X = X x fatorial (X - 1), se X <> 0
```

- **Programação funcional**

```
DEF fatorial;

    fatorial n = 1, n = 0

    fatorial n = n * fatorial
```

- **Programação imperativa**

  Pode ser definido dessa maneira:

```
function fatorial (n : integer) : Integer;

var f : integer;

begin

     f:= 1;

     while n > 0 do

     begin

          f := f * n;
          n := n-1;

     end;

    fatorial := f;

end;
```

ou dessa maneira:

```
function fatorial (n : integer) : Integer;

var f : integer;

begin

      if n = 0

         then f := 1

      else f := n * fatorial (n - 1);
```

```
            fatorial := f;

    end;
```

Constata-se porque é dito que a PF está tão próxima da notação normal da matemática e porque é mais fácil de ser entendida por um aluno iniciante, do que a programação imperativa.

3.3.1 Paradigmas de programação

Paradigmas de Programação:

- Programação Imperativa
- Programação Funcional
- Programação Declarativa
- Programação Orientada a Objeto (OOP)
- Programação orientada a eventos

Programação Imperativa

Consiste em definir procedimentos para executar seqüências de ações, isto é, seqüências de atribuições ou avaliações (ex: C, Pascal, Fortran, etc).

Programação Funcional

Estilo de programação, sem atribuição, nem alteração dos argumentos de funções. As funções se limitam a produzir novos valores. Neste paradigma de programação, qualquer função da linguagem é considerada uma função matemática pura que, para os mesmos argumentos produz sempre os mesmos valores.

Nunca nada é destruído.

Exs:

1. Uma função que junta duas listas produz uma nova lista sem alterar as listas originais.

2. Uma função que muda o número de portas de um automóvel produz um novo automóvel.

Linguagens: Lisp, Haskell, Hugs, Miranda, ML, Clean e outras.

Programação Declarativa

O programador preocupa-se apenas com o significado declarativo do seu programa, sendo os aspectos procedimentais da execução do programa tratados automaticamente.

O programador descreve o problema a ser resolvido e esta descrição é usada para encontrar uma ou mais soluções ao problema.

Ex: Prolog - usado para resolver problemas que envolvem objetos, as suas propriedades e relações.

Programação Orientada por Objetos

O programa é organizado em função de objetos, que contêm não só as estruturas de dados (isto é os registros), mas também as funções que sobre eles agem. A comunicação entre objetos é feita através de mensagens (que ativam as funções de um objeto). Mensagens normalizadas permitem que diferentes classes de objetos possam responder aos mesmos tipos de mensagens. Os objetos são normalmente organizados numa rede podendo os objetos mais especializados herdar as propriedades (i.e. funções e campos) dos objetos mais genéricos (herança). Este paradigma está particularmente bem adaptado à construção de interfaces gráficas.

São exemplos clássicos de linguagens com este paradigma o SMALLTALK e o SIMULA. O C++ e o JAVA são linguagens que utilizam o paradigma da programação orientada por objetos, embora também deixem ao programador a possibilidade de ignorar o paradigma da programação orientada por objetos e utilizar apenas o paradigma da programação imperativa. Por esta razão, estas linguagens são por vezes chamadas de linguagens híbridas entre linguagens imperativas e linguagens orientadas por objetos.

Programação Orientada a Eventos

Baseia-se na inexistência de um algoritmo principal que, em uma programação tradicional (imperativa), corresponde a um programa com início e fim. Em um programa orientado a eventos, tudo o que tem-se é uma forma que contém vários controles e a cada ação do utilizador (a que denominamos evento), o programa responde com a execução de um procedimento. Linguagens: Visual Basic, Visual C++, Macromedia Director, Java.

3.4 Características da Programação Funcional

Uma linguagem de programação nada mais é do que uma forma de comunicação entre o usuário e o computador, sendo necessário que a linguagem seja capaz de compreender tanto a máquina como o usuário.

PF, também chamada programação aplicativa, representa para pessoas de diferentes formações, idéias às vezes bastante diferentes, centradas no princípio básico de programação usando funções e aplicações. As linguagens de programação que tem com base estas duas idéias são denominadas "linguagens funcionais". Mas há bem mais de uma versão do que o nome PF realmente significa. Usando apenas definição de funções, como na matemática, e aplicação destas funções a argumentos, como mecanismos, o paradigma funcional exclui atribuição e controle como elementos de programação [36].

PF, como o próprio nome diz, é o tratamento de funções para resolução de problemas, sendo que essas funções podem ser passadas como argumentos para outras funções e retornando como o resultado da função. É utilizada uma notação normal como a matemática que se estuda no dia a dia.

É uma metodologia de programação que trata a computação como uma avaliação de funções matemáticas, onde uma função, neste sentido, pode ter ou não ter parâmetros e um simples valor de retorno.

Os parâmetros ou argumentos, como às vezes são chamados - são os valores de entrada da função, e o valor de retorno é o resultado da função.

A definição de uma função descreve como a função será avaliada em termos de outras funções. Por exemplo, a função $f(x) = x^2 + 2$ é definida em termos de funções de potenciação e adição.

Do mesmo modo, a linguagem deve oferecer funções básicas que não requerem definições adicionais. As funções podem ser manipuladas em uma grande variedade de formas em uma linguagem de PF e são tratadas como valores de primeira importância, o que é o mesmo que dizer que funções podem ser parâmetros ou valores de entrada para outras funções e podem ser os valores de retorno ou saída de uma função.

As funções nas linguagens funcionais podem ser nomeadas, como em outras linguagens, ou definidas anonimamente (algumas vezes durante a execução do programa) usando uma abstração lambda e usadas como valores em outras funções.

As linguagens funcionais também permitem que funções sejam do tipo "curry".

"Currying" é uma técnica para reescrita de funções com múltiplos parâmetros como a composição de funções de um parâmetro. A função do tipo "curry" pode ser aplicada apenas a um subconjunto de seus parâmetros. O resultado é uma função onde os parâmetros

neste subconjunto são agora fixados como constantes, e os valores do resto dos parâmetros ainda não são especificados. Esta nova função pode ser aplicada aos parâmetros restantes para obter o valor da função final.

Por exemplo, uma função adiciona (x,y) = x + y pode ser do tipo Curry de forma que o valor de retorno adiciona (2) - note que não há um parâmetro y - será uma função anônima, o que é equivalente à função adiciona 2(y) = 2 + y. Esta nova função tem apenas um parâmetro e corresponde a adicionar 2 a um número. Novamente, isso é apenas possível porque as funções são tratadas como valores de primeira importância.

Pode-se pensar na PF como simplesmente avaliação de expressões. O programador define uma função para resolver um problema, e passa esta para o computador avaliar, sendo que esta pode envolver várias outras funções em sua definição. O computador funciona então como uma calculadora que avalia as expressões escritas pelo programador através de simplificações até chegar a uma forma normal. A característica que domina na PF é que o significado de uma expressão é seu valor, e o papel do computador é simplesmente obtê-lo [7].

Outra característica é que uma função em uma linguagem funcional pode ser construída, manipulada e resolvida, como qualquer outro tipo de Expressão matemática, usando leis algébricas. Como a entidade principal da PF é a função, podem ser definidas listas de funções, funções podem devolver como resultado outras funções e podem ser passadas como argumento para funções.

Saindo do paradigma procedural de resolver as coisas e passando para uma filosofia de mudança do pensamento, na PF, tem-se uma maneira diferente de abstrair a solução dos problemas utilizando a habilidade humana para desenvolver funções, aplicar funções e conhecer o comportamento das funções na máquina para que tragam o resultado esperado. Onde um programa de computador é visto como uma função, a qual deve ser aplicada a um conjunto de dados de entrada para produzir-se um conjunto de dados na saída, que é o resultado da avaliação da função (que é a tarefa de calcular o valor da função), correspondente a execução do programa.

No estilo de PF, o programador não tem que se preocupar com:

- alocação de memória; não existe o comando de atribuição;
- valores não são atribuídos a localizações de memória, mas o programador dá nomes a valores que são utilizados em expressões; esses nomes podem ser utilizados em outras expressões ou passados como parâmetros de funções e
- os controles são realizados por função, recursão e expressões condicionais para

representar os problemas, não usando atribuições, podendo as expressões ser avaliadas a qualquer hora.

É um estilo de programação que enfatiza a avaliação de funções em vez da execução de comandos.

Programar em uma linguagem funcional consiste em construir definições e usar o computador para avaliar expressões. O papel principal do programador é o de construir uma função para resolver um problema dado. Essa função que pode envolver um número subsidiário de funções é expressa em uma notação que obedece a princípios matemáticos. O principal papel do computador é agir como um avaliador ou calculador: é seu trabalho avaliar as expressões e imprimir os resultados. O que distingue uma calculadora funcional das outras é a habilidade do programador de fazer definições para aumentar o poder de cálculo. Expressões que contém ocorrências de nomes de funções definidas pelo programador são avaliadas usando as definições dadas como regras de simplificação (ou redução) para converter expressões em um resultado.

Nos últimos anos a PF vem marcando presença nos cursos de Ciência da Computação em todo o mundo [55]. Ela tem sido usada com sucesso em uma grande variedade de projetos, e uma quantidade enorme de implementações de linguagens funcionais robustas e eficientes foram desenvolvidas. Justifica-se o estudo da PF por ela envolver notações e conceitos familiares para qualquer pessoa que tenha um conhecimento básico de matemática, dessa maneira sendo de fácil compreensão por estudantes que não possuem um contato amplo com linguagens de programação.

3.5 Trabalhando com a Programação Funcional

Pode-se entender o computador, de uma forma simplificada, como uma máquina capaz de:

a) avaliar expressões escritas segundo regras sintáticas bem definidas, como a das expressões aritméticas, tão bem conhecidas (3 + 5 - 8) obedecendo à semântica das funções primitivas das quais ela é dotada (por exemplo, as funções aritméticas básicas como somar, subtrair, multiplicar e dividir) e,

b) aceitar a definição de novas funções e posteriormente considerá-las na avaliação de expressões submetidas à sua avaliação. Para exemplificar uma interação com o computador, denominaremos o computador de máquina funcional.

Exemplo de uma interação com a Máquina Funcional.

usuário : (8 + 2) / 2
sistema: 5
usuário: 8 + 2 / 2
sistema: 9
usuário: f x y = (x + y) / 2
sistema: definição de f foi aceita
usuário: (f 3 5) + (f 10 40)
sistema: 29

Nas primeiras duas interações podemos observar que o usuário descreveu uma expressão aritmética e que o sistema avaliou e informou o resultado. Na terceira interação o usuário descreve, através de uma equação, uma nova função, que ele denominou de f e que o sistema acatou a nova definição. Na quarta interação o usuário solicita a avaliação de uma nova expressão aritmética usando o conceito recentemente definido e que o sistema faz a avaliação usando corretamente o novo conceito.

Também se pode com a máquina funcional avaliar uma expressão passando-a para o computador que irá reduzi-la a uma resposta das seguintes formas:

Supondo que o símbolo $(>)$ representa o símbolo de um interpretador, tem-se:

$>$ 30
30

O computador então devolve o número 30, pois ele é uma expressão em sua forma mais simples, não podendo sofrer um processo de avaliação.

Uma expressão mais complexa seria:

$>$ 4 * 3
12

Aqui o computador simplifica a expressão realizando a multiplicação.

Outra característica muito importante da PF é a construção de definições.

Uma lista de definições chama-se “script”.

Exemplo de script:
square x = x * x
somaEmult x y = 5 * (x + y)

No “script” acima duas funções, “square” e “somaEmult” foram definidas. A função “square” toma um valor x e o multiplica por ele mesmo. A função “somaEmult” devolve a soma de seus argumentos multiplicada por 5.

As funções “square” e “somaEmult” foram definidas através de equações e uma definição pode ser considerada como uma definição lógica de como a função se comporta. Em outras palavras, uma definição funcional pode ser vista como uma especificação do problema. Tendo criado scripts pode-se usá-los em uma avaliação na máquina funcional.

Exemplos:
> square (3 + 3)
36
> somaEmult 1 3
16
> square (somaEmult 2 4)
324

O objetivo de uma definição é ligar um nome a um valor. No “script” anterior, por exemplo, o nome “square” é associado a uma função que eleva um valor ao quadrado. Para serem avaliadas as expressões não necessariamente precisam que o programador crie um “script”.

Por exemplo, pode-se supor que os operadores aritméticos básicos são dados como primitivos, da linguagem. Outros operadores especiais podem aparecer em bibliotecas de funções pré-definidas. O programador pode reutilizar as definições existentes para construir outras em um “script”.

Por exemplo, as definições:
valor1 = 12
area = square valor1
introduzem duas constantes numéricas, sendo que a definição de área depende da definição da função square.
> area
144

Capítulo 4

LISP utilizado na educação

“Para mim, o essencial da investigação em I.A., reside na tentativa de compreender a própria natureza da inteligência natural”. (Seymor Papert)

4.1 Introdução

Como LISP foi uma das linguagens precursoras da PF e teve vários dialetos, vamos aqui nesse capítulo explorar um pouco de sua contribuição para a educação, para facilitar o entendimento do leitor e também descrever uma grande implementação na educação: a linguagem LOGO.

Verificando o porque ser mais fácil ensinar os alunos nesse paradigma funcional, por ter o raciocínio bem próximo ao raciocínio normalmente utilizado na matemática e por não ter que conhecer ou preocupar-se em com o funcionamento interno do computador.

4.2 O início das linguagens funcionais

Lisp foi originalmente pretendida como um modelo computacional de processos matemáticos, refletindo o rigor da matemática por si só [25]. É uma das linguagens mais antigas ainda em uso. Inicialmente, trabalhando em um problema de geometria, McCarthy e seus alunos desenvolveram uma ferramenta, que foi chamada FLPL (FORTRAN List_Processing Language), escrevendo um conjunto de sub-rotinas de processamento de listas, em Fortran, essas primitivas básicas de processamento de listas foram incorporadas a Lisp.

O uso de FLPL resultou na introdução da expressão condicional, possibilitando compor funções IF com funções de processamento de listas para serem usadas em programas mais complexos. Em 1958, McCarthy começou a usar a recursão, em conjunção com expressões condicionais, em sua definição de processamento de listas e convencendo-se da importância dessa combinação, que não era possível de ser utilizada em Fortran, ficou claro que uma nova linguagem era necessária, assim começou LISP.

Inicialmente, não havia preocupações com a IA, que precisava de representação para as inter-relações entre informações e dados, porém, um ponteiro e estruturas de listas vinculadas, são métodos naturais de estruturar dados e a IA ficou de olhos abertos. LISP permite novas maneiras de se pensar sobre programação, não na execução explícita de comandos seqüenciais e sim a partir de definição e composição de funções.

É uma linguagem favorita para programas em IA por duas razões principais: primeiro é altamente flexível, isto é, é possível escrever um programa LISP para produzir comportamento virtual para o computador, e segundo é indefinidamente extensível, o que significa que se o programador, sente que em Lisp falta alguma facilidade, pode escrever um programa em Lisp para prover essa facilidade [25]. Esse programa pode tornar-se parte integrante do Lisp pessoal do programador, por isso o fato de Lisp ter tantos dialetos. (MACLISP, INTERLISP,FRANZLISP, MULISP, SCHEME, COMMOM-LISP, CLOS).

Em seu livro: “The Little Lisper”, Friedman [21] enfatiza que Lisp é uma linguagem dominante para muitos trabalhos de IA tais como: robótica, reconhecimento de padrões, sistemas especialistas, resolução de problemas, prova de teoremas, manipulações algébricas e outras.

As listas, introduzidas em Lisp no começo dos anos 60, marcaram tanto a linguagem que, para muitos, Lisp era processamento de listas e só [36].

O conceito de lista é fundamental em PF. No caso de Lisp, por exemplo, determinou toda a estrutura da linguagem e em outras linguagens mais modernas como KRC, era o único meio de estruturar dados.

Em IAS tenta-se reproduzir mecanismos do raciocínio. Ora o raciocínio ocorre como uma seqüência de pensamentos, isto é uma lista e não é difícil compreender uma linguagem que usa listas como estrutura de dados fundamental [5].

Para a representação dos dados em Lisp, embora se utilizem principalmente às listas, que são as responsáveis por grande parte do seu poder, podem ser usados strings e arrays. Existem várias funções para construir e manipular listas, permitindo um nível de abstração maior do programa. Representa funções através do processamento de listas.

A máquina universal de Turing é uma máquina que pode simular qualquer outra máquina de Turing que se possa descrever, McCarthy fez exatamente isso com Lisp, es-

creveu um interpretador Lisp em Lisp.

A função universal foi denominada de Expressão-S, S significava simbolic, language (linguagem simbólica), foi traduzida para assembler e colocada com as sub-rotinas de manuseio de listas, sendo assim desenvolvido o primeiro sistema LISP. Tanto os programas como os dados são estruturados em forma de listas [40].

4.3 Pontos fortes do LISP

O forte do LISP é a manipulação simbólica, não apresentando recursos competitivos na manipulação de estruturas matriciais. Curiosamente LISP apresenta, no campo numérico, um recurso que causa inveja a qualquer FORTRAN ou outra linguagem mais orientada ao campo numérico. É a precisão "infinita". LISP pode efetuar cálculos com qualquer precisão, limitada apenas pela capacidade da memória disponível [47].

Cabe destacar que, ainda que associado tradicionalmente à IA, a linguagem Lisp tem certas características que facilitam a tarefa do programador em aplicações muito variadas. Por exemplo, em áreas como o desenho de circuitos integrados VLSI, o processo de sinais digitais, o ensinamento da informática, o desenvolvimento de sistemas informáticos (como sistemas operacionais, compiladores, interpretes e programas de utilidades diversas), ou o manejo de símbolos matemáticos, os padrões de trabalho LISP podem chegar a competir (ou competem) em eficiência com os sistemas baseados em Pascal ou Fortran.

A origem híbrida de LISP envolvendo resultados matemáticos tais como cálculo-lambda e funções recursivas bem como soluções elegantes de implementação, para fazer a linguagem rodar em máquinas com memórias tão limitadas como 4Kb de computadores dos anos 60 levou a que muitos olhassem Lisp como um desafio matemático e não como uma ferramenta de desenvolvimento [5]. Conseguir olhar Lisp como algo realmente para resolver problemas e não como uma implementação de cálculo-lambda não foi fácil sendo esta ainda a visão de muitos atualmente.

Maurer [34] em seu livro: "The Programmer's Introduction to Lisp", caracteriza Lisp como uma linguagem funcional, como uma linguagem simbólica, como uma linguagem de processamento de listas, como uma linguagem recursiva e como uma linguagem lógica.

4.4 Uma implementação na educação

LOGO é uma palavra derivada do grego logos, que contém ao mesmo tempo a noção de logo-razão, logo-linguagem e logo-cálculo [8].

Em informática, é uma linguagem de programação que foi desenvolvida nos EUA, usada freqüentemente para ensinar crianças, desenvolvida originalmente por Seymour Papert, MIT, cujas raízes derivam de Lisp. Durante esse tempo passou por diversas fases: foi concebida nos anos 60, gestada nos anos 70, caminhou vacilantemente pelos anos 80, atingiu a maioridade nos anos 90.

Desde de sua criação até 1976, LOGO ficou restrito a estudos e aplicações de laboratórios, tais como: o MIT, O Departamento de Inteligência Artificial da Universidade de Edinburgh e o Instituto de Educação da Universidade de Londres.

Com o projeto "An Evaluative Study of Modern Technology in Education", começado em 1977 na Escola Pública de Brookline, usando LOGO em um micro-computador 3500 (criado por Marvin Minsky), aplicado em 16 alunos da 6^a série, pode-se dizer que o LOGO começou a sair dos laboratórios e penetrar na escola.

Um dos elementos facilitadores para a penetração do LOGO em outros centros de pesquisa e nas escolas foi o desenvolvimento dos micro-computadores da Texas Instrument e da Apple.

A princípio não houve preocupação com o papel do professor no ambiente LOGO. Papert no livro Logo: Computadores e Educação [44], deixa transparecer que "idéias poderosas" surgiriam espontaneamente da atividade do aluno ao programar em Logo e isso aconteceria sem uma maior intervenção do professor, cabendo-lhe apenas auxiliar os alunos no que diz respeito à sintaxe do LOGO.

O objetivo geral do trabalho com LOGO é criar condições para o desenvolvimento do raciocínio lógico-dedutivo, facilitando a expressão da criatividade, o trabalho cooperativo e a postura crítica.

4.4.1 LOGO na educação

LOGO é uma linguagem estruturada voltada para a educação, que tem como objetivo permitir que uma pessoa se familiarize, através do seu uso, com conceitos lógicos e matemáticos. O ponto forte da linguagem é sua capacidade gráfica. O primeiro passo para utilização da linguagem é, normalmente, a familiarização com algumas palavras do vocabulário Logo e a exploração com estas palavras da criação de desenhos na tela do computador.

Na realidade, LOGO pode ser vista muito mais como uma filosofia de educação do que como uma linguagem de programação. Por isto mesmo, existem diversas implementações diferentes de Logo, algumas inclusive em português, cada uma delas com características peculiares.

A filosofia surgiu com base nas referências teóricas sobre a natureza da aprendizagem desenvolvida por Piaget (reinterpretadas por Papert), e nas teorias computacionais, principalmente a da IA, vista como Ciência da Cognição, que para Papert também é uma metodologia de ensino-aprendizagem, cujo objetivo é fazer com que as crianças pensem a respeito de si mesmas.

Até mesmo instruções básicas, como, por exemplo, à função para limpar a tela, variam de implementação para implementação. De qualquer modo, estas diferenças não invalidam cada uma destas implementações, que também têm muito em comum, sendo a tartaruga uma unanimidade.

Outra característica marcante do LOGO é a possibilidade de incorporar novos comandos ao seu vocabulário e como cada palavra designa uma função, programar em LOGO é definir funções novas a partir de funções primitivas usando composição e recursividade. Por exemplo, pode-se “ensinar” ao computador que o comando quadrado será equivalente à repetição de quatro vezes dos comandos de deslocamento de um número fixo de unidades e de rotação de 90 graus.

Visualiza-se na figura abaixo 4.1 um desenho realizado em um projeto com LOGO:

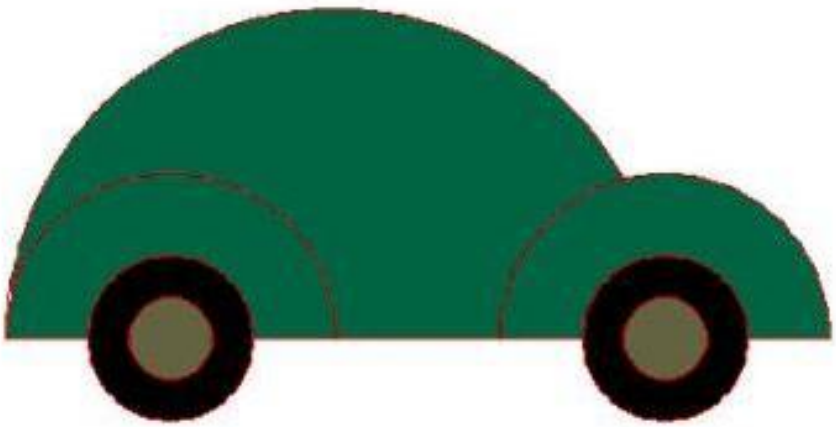

Figura 4.1: Desenho de um carro feito em um projeto, utilizando Logo

Nessa outra figura 4.2 um desenho de um outro projeto com LOGO:

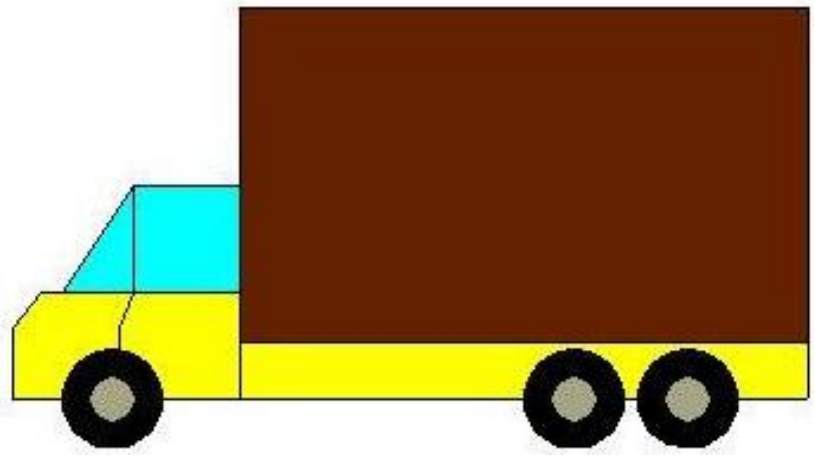

Figura 4.2: Desenho de um outro tipo de carro feito em um projeto, utilizando Logo

E nesta outra figura 4.3 um desenho de um boneco, também realizado com LOGO:

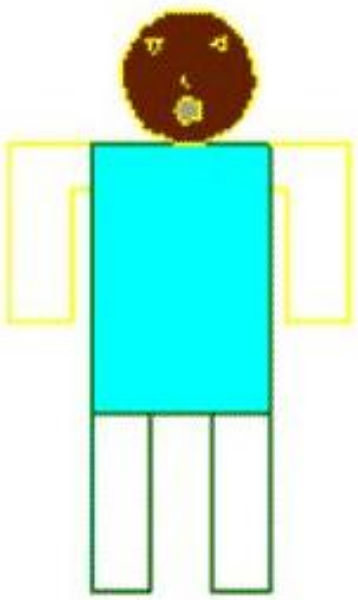

Figura 4.3: Desenho de um boneco feito em um projeto, utilizando Logo

O conjunto de comandos constitui um micro-mundo, que na filosofia Logo, é a base para a mudança de padrões de acesso ao conhecimento, na medida em que o próprio aluno é capaz de criar um micro-mundo onde seus pensamentos são expressos através de palavras e desenhos.

Para Meira [35] a criança está no controle: a criança programa o computador. E ao ensinar ao computador como pensar, as crianças embarcam numa exploração sobre como

elas próprias pensam. A experiência pode ser inusitada: pensar sobre o pensamento faz da criança um epistemólogo, uma experiência nem sempre ao alcance da maioria dos adultos.

4.5 A interação com o meio do aprendizado

A linguagem LOGO permite à criança agir sobre o mundo exterior a partir de seus próprios modelos de pensamento [8].

A visão que Papert tem do homem e do mundo situa-se numa perspectiva interacionista, sendo o conhecimento o produto dessa interação, que é centrada nas formas com que o mundo cultural age e influencia o sujeito em interação com o objeto. Ao contrário de Piaget, Papert enfatiza que aquilo que aprendemos e o como aprendemos depende dos materiais culturais que encontramos à nossa disposição.

De acordo com sua teoria do conhecimento e do desenvolvimento humano, o processo educacional tem como pressuposto que o aluno não aprende apenas pelo ensino formal e deliberado, que é um aprendiz inato, que mesmo antes de chegar à escola apresenta conhecimentos adquiridos por meio de uma aprendizagem natural, espontânea e intuitiva, que se dá através da exploração, da busca e da investigação, a qual pode ser caracterizada como uma real auto-aprendizagem.

Aquilo que o aluno aprendeu porque fez, após ter explorado, investigado e descoberto por si próprio, além de contribuir para o desenvolvimento de suas estruturas cognitivas, reveste-se de um significado especial que ajuda a reter e transferir com muito mais facilidade aquilo que foi aprendido.

LOGO designa simultaneamente uma teoria de aprendizagem, uma linguagem de comunicação e um conjunto de unidades materiais que permite demonstrar os processos mentais empregados por um indivíduo para resolver os problemas que se lhe apresentam e aos quais ele propõe uma solução, num contexto de ação sobre o mundo exterior [8].

Está imbuída na filosofia do LOGO, como a concebeu Papert, a idéia que a aquisição de um conhecimento não se dá em função do desenvolvimento, mas principalmente na maneira pela qual as pessoas se relacionam com o meio, ou seja, as condições que este oferece para exercitar o pensamento qualitativo. Acredita na necessidade da pessoa controlar sua aprendizagem, poder reconhecer e escolher entre várias possibilidades de pensamento estruturado.

Portanto, com esses pressupostos, é de fundamental importância oferecer à criança condições de fazer manipulações intuitivas, confrontar e filtrar suas intuições.

Um outro aspecto importante nas concepções de Papert é o fato de no LOGO se considerar o erro como um importante fator de aprendizagem, o que oferece oportunidades para que o aluno entenda porque errou e busque uma nova solução para o problema, investigando, explorando, descobrindo por si próprio, ou seja, a aprendizagem pela descoberta.

Os procedimentos de análise e correção no processo de aprendizagem pelo LOGO possibilitam a descoberta de diferentes caminhos na solução de problemas, sendo que esses caminhos advêm de um contexto cultural onde não há certo e errado, pois as soluções são pessoais.

4.6 O aprendizado com a linguagem LOGO

Permite introduzir as crianças na geometria como algo que elas podem explorar através de seus próprios movimentos.

É esse tipo de aprendizagem que a filosofia do ambiente LOGO pretende que seja desenvolvida com a ajuda da linguagem de programação LOGO, que possibilita integrar habilidades corporais com as intelectuais, a visualização da representação do modo como pensamos, promovendo o desenvolvimento do pensamento estruturado, modular.

O computador pode concretizar e personalizar o formal e sendo bem utilizado permite abordar de forma concreta os conhecimentos até então somente acessíveis vias processos formais, o que permitiria transpor o obstáculo na passagem do pensamento concreto para o abstrato (identificação de Piaget) [44].

Foi considerada bastante arrojada, colocando a criança como agente programador, pois todos os programas para computador até então, enquadrava-se na linha estímulo-resposta e também colocando à disposição recursos gráficos pouco comuns e usados quase que exclusivamente no meio computacional, abstraindo o uso do computador na figura de uma “tartaruga”.

LOGO é na realidade uma linguagem de programação de grande escala. Isto significa que é quase infinitamente flexível. Como a maioria das linguagens de programação convencionais, pode ser utilizada como um modelo para uma ampla variedade de fenômenos, tanto naturais quanto artificiais [23]. Diferente da maioria das linguagens de programação, LOGO tem a facilidade de aprendizagem e de uso como seus principais objetivos. Com LOGO, é muito fácil programar coisas interessantes, que nunca poderíamos esperar realizar sem um computador dentro de um período muito curto, a partir do início do aprendizado de uma linguagem.

A teoria do conhecimento adotada por LOGO faz uma síntese entre a concepção de

Piaget sobre o desenvolvimento da criança e o estudo, em IA, do problema do pensamento. A criança não é mais um “objeto” a ser modelado, educado. Ela torna-se “sujeito” [8].

LOGO é uma linguagem verdadeiramente interativa, por permitir que a criança comande suas ações e receba respostas imediatas. Ao trabalhar com a linguagem LOGO, o erro é tratado como uma tentativa de acerto, ou seja, uma fase necessária à nova estruturação cognitiva. As respostas mencionadas aos comandos são direcionadas ao estímulo para uma nova tentativa. Esta linguagem desafiadora pode ser usada por alunos de todas as idades, ou por qualquer usuário interessado em “criar e construir o seu conhecimento”.

Tanto no ambiente LOGO quanto na linguagem do mesmo nome há preocupação de amparar o aluno no momento do erro e fazer com que perca o medo de cometê-lo, perca o medo de ser punido ou humilhado por tê-lo cometido. Mas tal altitude não é tomada apenas em função de algum doce humanismo; ela tem uma sustentação psicológica: a criança aprende com seus erros; o erro é parte integrante do processo de aprendizagem e, portanto, longe de ser banido, deve ser, pelo contrário, trabalhado [28].

Em suas diversas explorações dos comandos primitivos e uso do equipamento a criança é estimulada a desenvolver projetos com a proposta de ensinar a tartaruga. Assim, inverte-se a situação da criança ser ensinada pela máquina, para o completo domínio sobre o computador [50].

Ve-se um exemplo na figura 4.4 de um projeto de um jogo, o qual consiste que a criança adivinhe o número que a tartaruga está pensando. Assim, ao dar um número compreendido entre 0 e 100, a tartaruga responderá se o número a ser adivinhado é maior ou menor, forçando a criança a compreender qual é o número maior e menor, fazendo-a perceber a sua ordem, na seqüência numérica.

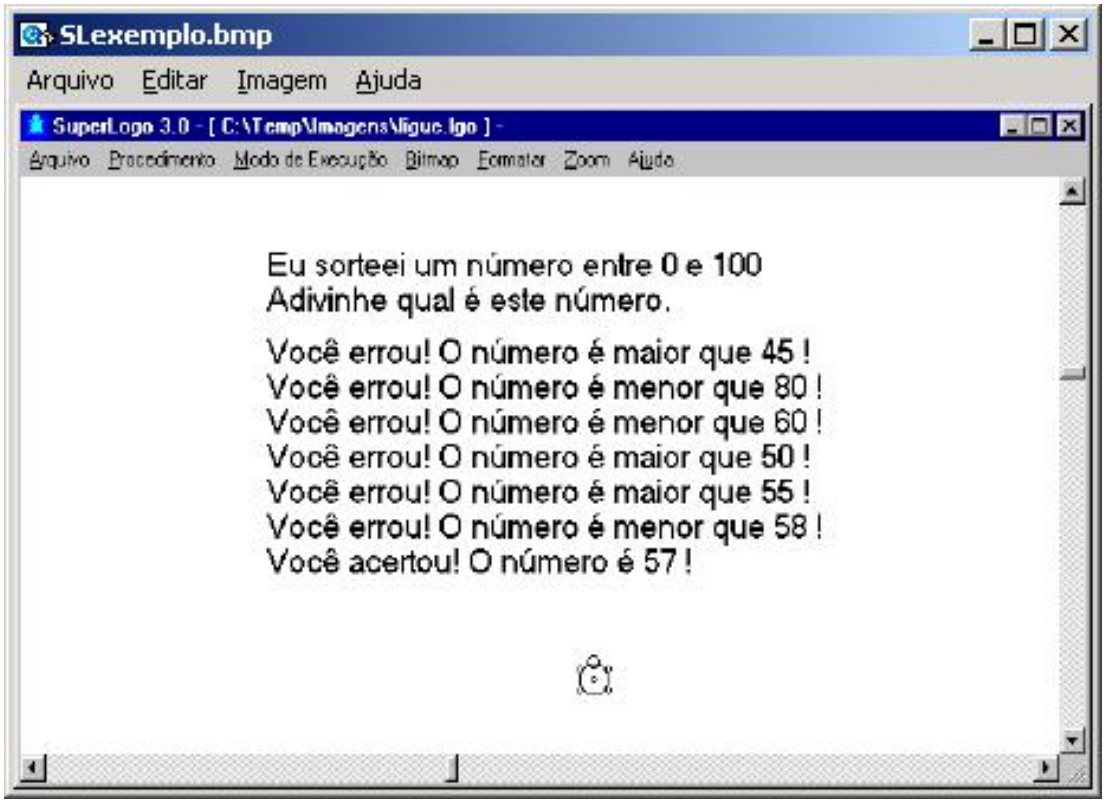

Figura 4.4: Exemplo de um projeto em Logo

A análise da atividade de programar o computador, usando uma linguagem de programação como o LOGO gráfico, permite identificar diversas ações, que acontecem em termos do ciclo descrição-execução-reflexão-depuração-descrição, que o aluno realiza e são de extrema importância na aquisição de novos conhecimentos [57].

O uso do programa LOGO na construção de dispositivos permite a exploração de conceitos de distintas áreas de conhecimento, favorecendo a aprendizagem de novos conceitos. As atividades se caracterizam por propiciarem aprendizagem através de “design”, pois exigem o emprego da heurística na solução de problemas de definição nem sempre muito claras e com delineamento difuso, semelhantes aos problemas enfrentados nos incidentes do dia-a-dia. A solução encontrada é aceita quando o dispositivo apresenta o comportamento desejado pelo usuário que o projetou, o que geralmente ocorre após um processo dinâmico de ação-reflexão-depuração, tanto nos aspectos relacionados à construção do mecanismo, como nos comandos e na lógica do programa LOGO, conforme apontado em Valente [59].

4.6.1 Funcionamento do LOGO

A linguagem LOGO contém dois aspectos importantes que são: os gráficos e os relacionados à manipulação de palavras e processamento de listas de comandos. Em seu aspecto gráfico, assim que a linguagem LOGO é carregada na memória do computador, aparece na tela do monitor um cursor em forma de tartaruga ou um triângulo. Daí que para se ter desenho na tela é dado instruções específicas à tartaruga, através de comandos digitados no teclado que especifica o que a tartaruga deve fazer.

Essa tartaruga que é capaz de andar pela tela deixando seu rastro, desenhando ou executando qualquer outra ordem, é uma metáfora para a atividade de programar. O universo material hardware LOGO é uma reunião de módulos tecnológicos [8]. Ele comporta atualmente uma tartaruga mecânica capaz de:

- mudar sua posição;
- mudar sua direção;
- deixar no solo um traço de seu deslocamento ou não;
- emitir um sinal sonoro ou luminoso;
- testar seu contato com um obstáculo.

Ensinar a tartaruga a obedecer às ordens é saber exatamente o que e como essa ordem deve ser executada, o procedimento que a criança cria para ensinar a tartaruga deve conter todos os passos que a tartaruga deve executar para conseguir o resultado desejado. Para ensinar a tartaruga, o aluno precisa ter estratégias e aprender brincando, os conceitos que deseja passar para a tartaruga.

Ao mesmo tempo em que a criança está desenvolvendo a construção de seu conhecimento, também tem que interagir com o computador, através da escrita e obedecer a certas regras para poder ensinar a tartaruga na linguagem que ela entende, organizando dessa maneira um pensamento formal.

Os conceitos espaciais são usados para comandar a tartaruga, que se movimenta em atividades gráficas , que envolvem conceitos espaciais que estão presentes nos alunos a nível intuitivo.

O uso do LOGO resgata a aprendizagem construtivista, provocando uma mudança com ênfase na aprendizagem ao invés de colocar no ensino; na construção do conhecimento e não na instrução.

A metodologia que Papert propõe é carregada de significado lúdico, proporciona à criança uma situação de brinquedo. O diálogo que se estabelece com a máquina (tartaruga) é naturalmente, uma atividade de brincadeira, em que a criança aos poucos é levada a aprender as noções básicas do sistema LOGO. Ao brincar de tartaruga, a criança, e até mesmo o adulto, projeta-se nas ações baseadas na própria experiência de deslocamento no espaço, as quais são similares as da tartaruga da tela. Há aqui um deslocamento dos significados da ação da tartaruga-criança para a tartaruga-da-tela.

O computador pode entender instruções que a criança dá a ele em LOGO porque tem um programa poderoso chamado interpretador LOGO. Este interpretador age como esperaríamos que um intérprete humano agisse. Recebe a instrução LOGO da criança (digamos PARAFRENTE 50) e traduz em código binário de 1s e 0s que os circuitos do computador entendem. E um interpretador LOGO pode não só entender as instruções de LOGO fundamentais embutidas como PARAFRENTE, PARADIREITA, PARATRAS e PARAESQUERDA, mas também tem o poder de construir um dicionário de novas palavras LOGO quando e como foi definido pelas crianças (por exemplo, DESENHE-UM-QUADRADO).

A recursão é um procedimento que chama a si mesmo. Assim como vários espelhos que reproduzem a mesma imagem. Brincando, por exemplo, com o comando repita, podemos ensinar a tartaruga a fazer repetidas vezes o mesmo desenho em diferentes posições, usando a recursão [50].

Papert [44] salienta que de todas as idéias que apresentou às crianças, a recursão se destacou como uma idéia de provocar uma resposta entusiástica. Ele acha que isso acontece em parte porque a idéia de continuar indefinidamente toca fundo nas fantasias de qualquer criança e também porque a recursão tem suas raízes na cultura popular. Há, por exemplo, a charada da recursão: “Se você tem dois desejos, qual é o segundo?” (mais dois desejos).

Com recursão, LOGO deixa de ser “coisa de criança” e passa a permitir o desenvolvimento de projetos mais elaborados. É importante deixar claro a distinção entre um processo recursivo e um processo repetitivo obtido através do comando repita. O comando repita é geralmente utilizado quando se conhece a priori o número de repetições. Recursão é mais útil quando queremos repetir algo até que uma determinada condição é satisfeita [58].

O LOGO também possui comandos para manipular palavras e listas (conjunto de palavras), com os quais é possível ensinar a tartaruga a produzir uma frase, criar histórias, integrar a parte gráfica com a manipulação de palavras para fazer animações, ou ainda explorar conceitos de Ciências, Física, Química e Biologia [46].

Existe ainda um segundo paradigma do qual LOGO se aproximou quando possibilitou o trabalho com múltiplas tartarugas: o paradigma orientado a objetos, representadas por formas, cores e comportamentos diferentes, mesmo que sejam dados a elas comandos idênticos.

Atualmente já existem versões de Logo orientado a objetos que integra mais essa característica à programação LOGO tradicional.

Há também, o paradigma da PF em LOGO, que é uma herança do Lisp, através da manipulação de listas, que é pouco compreendida pela comunidade de usuários LOGO. Lista é uma estrutura de dados que permite representar uma série ordenada de itens, que podem, por sua vez, ser listas. As primitivas para manipulação de listas, são funções.

Para aprofundamento em projetos LOGO consultar em Mendonça [38].

4.7 Facilidades das linguagens funcionais em relação às linguagens imperativas

Referindo-se ao exemplo citado no capítulo 3, sobre o fatorial, percebe-se que o caracteriza a PF é que não se raciocina com endereço de memória. Quando programamos com linguagens imperativas, tem-se o conceito de nome de um endereço de memória e o programa com esse nome de endereço, reserva certas posições de memória e depois manipula o conteúdo desse endereço.

Na linha de comando, retirado do exemplo de fatorial, tem-se:

f := f * n;

ou seja, coloca no endereço, ao qual foi dado o nome f, o conteúdo que ele tinha, multiplicado pelo conteúdo de n.

Em PF não se raciocina assim, raciocina-se com dados e resultados para a solução dos problemas.

Um problema se resume em dado um elemento do conjunto de dados de entrada, achar qual elemento do conjunto de resultados que corresponde.

Resolver o problema é implementar a função que associa os elementos do conjunto de entrada aos elementos do conjunto de resultados.

Capítulo 5

Desenvolvimento do protótipo

“Comece fazendo o que é necessário, depois o que é possível, e de repente você estará fazendo o impossível. ” (São Francisco de Assis)

5.1 Aplicando os conceitos de Programação Funcional

Os autores Rocha [48], Castro [11], Fernandes [19] e Rodrigues [49] propõem o emprego de novas metodologias e ferramentas de auxílio ao ensino de programação, incluindo em seus estudos algumas sugestões interessantes, como a aplicação do paradigma de programação declarativo pelo emprego de ferramentas da PF, inclusive o LOGO. O modelo tradicional de ensino de programação pelo paradigma imperativo é mantido, mas em virtude de atender a necessidade de melhorar ou desenvolver no aluno a capacidade de solucionar problemas, a PF é introduzida primeiro, com o objetivo de promover um nivelamento tornando o paradigma imperativo menos traumatizante e o aluno menos propenso a desistência.

A abordagem mais tradicional de ensino de programação, a programação imperativa, tem trazido sérios problemas para quem nunca programou, pois exige que se repense a forma de resolver um problema como uma “receita de bolo”, o que não é o que fazemos no dia-a-dia. Por isso a abordagem que vem ganhando força nos últimos anos é a que se baseia em PF, pois, dizem seus defensores, ela de início facilita um “nivelamento” entre os estudantes porque poucos tiveram a experiência de já ter programado usando tal formalismo, além de utilizar um raciocínio matemático, com o qual os estudantes calouros

já estão acostumados. Para uma primeira aproximação com a área de programação de computadores, o uso de linguagens procedurais exige um esforço cognitivo muito grande, tendo em vista que além de se criar o hábito de resolver problemas com o computador, o aluno precisa aprender um novo modelo computacional. O uso de PF reforça a utilização de um ferramental fundamental para a modelagem de problemas e concentra os esforços em uma abordagem mais rigorosa, facilitando inclusive a discussão introdutória de análise de programas.

Em primeiro lugar, estudantes tendem a pensar na máquina de modo antropomórfico, isto é, dotando-a de qualidades humanas, supondo que ela poderia entender comandos incompletos ou errôneos. A necessidade de descrever comandos com rigorosa precisão torna-se assim uma característica muito artificial e até mesmo antipática para o estudante.

São comuns também:

- as dificuldades no entendimento do significado de um comando enquanto componente de uma solução;
- a sintaxe do comando numa certa linguagem de programação;
- e o resultado da execução daquele comando pela máquina.

Essas dificuldades são encontradas durante a elaboração de um programa. Além de ter de encontrar comandos para a solução de um problema, o aluno deverá ordená-los corretamente para criar um programa e, finalmente, verificar e ajustar o funcionamento deste.

As linguagens funcionais estão próximas à matemática, sendo, portanto, um veículo ideal para o ensino dos princípios de programação.

Pode-se argumentar que a matemática também é tão difícil de aprender quanto escrever programas procedurais. A isto se pode contra-argumentar lembrando que para alunos de computação isso não se aplica, pois a matemática já é um formalismo incorporado à linguagem e o conceito de função é fundamental para a modelagem de problemas do mundo real.

Assim, ao invés de o aluno ter dois novos desafios, conhecer um modelo computacional e utilizar o computador, retarda-se um deles, visto que os primeiros passos serão dados usando um formalismo familiar.

Os currículos de computação que adotam a linguagem funcional para seus cursos introdutórios o fazem por diferentes motivos, dentre eles são destacados três atributos compartilhados por todas as linguagens funcionais:

1. são de nível mais alto que as linguagens de programação tradicionais;

2. são mais próximas da especificação; e

3. não têm efeitos colaterais na avaliação, ou seja, variáveis declaradas não mudam de valor durante a execução.

Além desses pontos, o uso de linguagens funcionais dispensa o programador de prescrever explicitamente o fluxo de controle.

5.2 O jogo, como meio facilitador do aprendizado

Oferecer um conjunto mais rico de materiais para o aprendizado e, com isso, contribuir significativamente para a exploração e pesquisa dos alunos é uma característica da informática na educação, com este trabalho pretende-se mostrar que o uso do paradigma funcional para o ensino de programação de computadores, pode tornar-se interessante, principalmente se aplicado em conjunto com técnicas de jogos, ou seja, através de uma atividade lúdica.

Algumas das razões para a utilização do jogo no desenvolvimento deste trabalho são:

1. O uso do jogo permite ao aluno realizar os mais diversos experimentos, permitindo que aprenda explorando e realizando as suas próprias experimentações através da simulação. Deve-se, no entanto, levar em conta que o modelo representa uma simplificação da PF, apresentando apenas o conceito do paradigma funcional, não contendo, então toda a complexidade e benefícios das linguagens funcionais;

2. O uso do jogo pode proporcionar uma interface que facilita a aprendizagem e a interação entre aluno e sistema. Este recurso é importante em um software educacional por proporcionar um ambiente motivador e agradável de aprender;

3. a união da simulação com o jogo proporciona ao aluno um tipo de aprendizado por exploração e descoberta, ou seja, o aluno tem plena liberdade de guiar seu próprio aprendizado, podendo realizar os experimentos que considerar mais interessantes;

4. gerar uma forma mais atraente de ensino, que se adapte as necessidades de cada aluno, onde este avance de acordo com o seu tempo de assimilação e disponibilidade, podendo rever determinado assunto quantas vezes forem necessárias;

5. interação com o aluno, fazendo com que o aluno consiga ver que ele consegue fazer o computador fazer o que ele quiser criando uma familiarização do aluno com os computadores em geral;

6. incentivar a capacidade de abstração de problemas e de sistematização do raciocínio das soluções para problemas;

7. possibilitar uma introdução menos traumática do aluno iniciante e inexperiente a um ambiente de programação;

8. fundamentar a capacidade de abstração e o exercício do raciocínio no aluno;

9. aprendizagem por tentativa e erro.

O mundo da tecnologia e da informação nos fornece antenas, aprimora os nossos sentidos, permite-nos viver em um bem-estar com que os nossos antepassados não ousariam sonhar. Um único luxo, porém, não nos é permitido: interromper os nossos processos de aprendizagem subtrair-nos à formação permanente. Antes a escola era treinamento para a existência, depois instrução e educação em vista do ingresso no mundo do trabalho. Agora é uma necessidade de vida, tanto quanto o ar que respiramos [31].

5.3 As figuras geométricas

Para a realização deste trabalho, foi escolhido trabalhar com elementos que já fossem conhecidos dos alunos, evitando assim uma maior carga cognitiva e simplificando a maneira de apresentar novos conceitos do paradigma funcional e raciocínio geométrico espacial de duas dimensões, ao aluno que está iniciando seus estudos com o auxílio de uma ferramenta tão poderosa, mas não tão inteligente como alguns costumam pensar a respeito do computador.

Então se resolveu trabalhar com as figuras geométrica mais simples e conhecidas de todos como: retângulo, retângulo arredondado, elipse e triângulo, para montar o protótipo que se chamou de: Mundo dos Blocos Geométricos.

Tendo o objetivo de mesclar a atividade lúdica, através de um jogo simples de solução de alguns problemas que serão incorporados ao protótipo com o ensino de conceitos e reflexões sobre o paradigma funcional e geometria espacial, utilizando-se da aprendizagem por ensaio e erro, aprendizagem por fazer e aprendizagem por descoberta, descritas no capítulo 2.

Em Barros [6] é mencionado que enquanto exigirmos que os alunos tenham postura ereta, levantem as mãos antes de responder e que mantenham carteiras enfileiradas em perfeita ordem, não podemos esperar que sejam pensadores divergentes, ativos. A atmosfera passiva e convergente jamais inspirará soluções incomuns, contos humorísticos ou o uso original de objetivos corriqueiros.

Portanto, há duas condições essenciais a qualquer programa destinado à criatividade:

1. Rejeitar a crença de que a criatividade é inata, um dom especial de poucos escolhidos.

2. Modificar o ambiente de sala de aula, adequando-a a busca de soluções divergentes.

É através desse pensamento de mudanças de ambiente, de mudanças da maneira tradicional de se ensinar, com aulas expositivas e maçantes, onde o aluno não participa do processo de construção do conhecimento, que se resolveu desenvolver um projeto que além de possibilitar o conhecimento do paradigma funcional e geometria espacial, torne o processo simples e intuitivo, onde o aluno vai tornar-se parte essencial no desenrolar da atividade, em um ambiente agradável, dinâmico e simples.

Aprendendo dessa forma PF, não da maneira tradicional de se ensinar uma linguagem, dentro dos moldes formais ou teóricos, mas aprendendo de maneira pragmática, lúdica e utilitária os conceitos inerentes à PF que estão sendo utilizados no jogo.

Como escreveu Huizinga [26]: *O homem é, sobretudo um animal que brinca. A atividade lúdica é importantíssima no aprendizado*.

Entende-se que o sucesso na aprendizagem de programação depende mais das habilidades de abstração e menos das habilidades de codificação.

Consciente da amplitude do tema houve necessidade de delimitar o conteúdo a ser enfocado no presente estudo. Pretende-se centrar o esforço deste trabalho nos mecanismos básicos da PF que são: funções, recursão e sequências.

5.4 Implementação: Mundo dos Blocos Geométricos

Dada a natureza do conteúdo a ser ensinado, o trabalho desenvolveu-se centrado na figura do aluno e sua relação com a máquina, explorando os conceitos de PF, visando o processo de desenvolvimento do pensamento dos alunos e as dimensões pedagógicas que envolvem o uso do computador na educação.

E com essa relação direta e criativa com a máquina, o aluno, vai se deparando com os obstáculos, buscando suas próprias soluções para os problemas apresentados, através da

brincadeira. A resolução do problema propicia ao aluno, a investigação e descoberta, pois através do jogo o aluno pode colocar a si próprio situações-problema, desafios, formular hipóteses e trabalhar sobre os seus erros de uma forma construtiva.

Através de movimentos básicos (primitivas), o aluno vai formando os desenhos para a solução da situação-problema que reflete o que o aluno está pensando, a maneira que ele está tentando construir seu pensamento. Formando dessa maneira, através dos movimentos o procedimento correto para solucionar o problema, privilegiando a autonomia, exploração, liberdade e criatividade.

Um grande número de teorias de aprendizagem hoje utilizadas são categóricas ao afirmar que a aprendizagem só ocorre com a experiência, é o aprender fazendo.

Resolveu-se fazer um programa que trabalha com os objetos (figuras) geométricas, que construirão outras figuras geométricas para dar ao aluno uma interface funcional, onde ele poderá visualizar e aprender os mecanismos básicos da PF no computador virtual.

Pretende-se, com esse mundo de blocos, sugerir uma mudança na nossa cultura educacional, que oferece poucos recursos aos alunos para que eles entendam o que estão aprendendo, e não utilizem apenas a memorização dos conceitos apresentados, mas enfatizem a construção de seu conhecimento.

Para Wygotsky [60], a aprendizagem mantém com o desenvolvimento uma relação dialética e pode orientar e estimular processos internos do desenvolvimento, ampliando o seu papel nesse processo. Apresenta uma posição inovadora no que diz respeito à relação entre desenvolvimento e aprendizagem, pois, partindo da noção de "zona de desenvolvimento proximal ", afirma que a aprendizagem desperta uma série de processos evolutivos internos, capazes de operar quando o indivíduo encontra-se em interação com seu ambiente sócio-cultural. Uma vez, que estes processos se internalizam, convertem-se em parte das conquistas desenvolvimentistas, determinando a unidade dos processos de aprendizagem e desenvolvimento.

A teoria da zona de desenvolvimento proximal define aquelas funções cognitivas que ainda não amadureceram, mas que estão em processo de maturação, relacionando o nível de desenvolvimento efetivo do aluno com a capacidade potencial de aprendizagem e enfatizando a importância da ajuda externa que pode ser do professor, ou do companheiro, ou ainda da manipulação de instrumentos.

Almeida [2] cita que: *Há um certo senso comum que diz: "A escola é velha, a informática é nova. Os professores são ultrapassados, os métodos tecnológicos são inovadores. Os alunos são desmotivados, a tecnologia dos botões e telinhas trará aos jovens o desejo de conhecer".*

Essas profecias tecnológicas simplificadoras esquecem-se de que o que determina a

eficácia do ensino e da aprendizagem é a existência de um plano pedagógico escolar adequado, rico, consistente, motivador, crítico e inovador.

As escolas que têm um plano pedagógico ruim usarão a tecnologia (qualquer que seja ela) para fazer o seu trabalho de forma ainda pior, pois a tecnologia não conserta nada, não inventa consistência para um programa de baixa qualidade educacional. Ela apenas potencializa o que existe.

Com esse pressuposto, acredita-se e sugere-se uma estratégia rica para construir o conhecimento, com essa ajuda externa, no papel do computador, que se pretende, através do jogo: mundo dos blocos geométricos, incentivar o aluno no seu processo de aprendizagem do tipo saber-fazer.

5.4.1 Interação com o ambiente desenvolvido

O Mundo dos Blocos Geométricos é um programa para auxiliar o entendimento dos mecanismos básicos de PF e noções de figuras geométricas planas de duas dimensões, dando suporte aos alunos para a construção do conhecimento, principalmente as crianças entre 7 e 10 anos e alunos iniciantes dos cursos de computação, que ainda não tiveram nenhuma experiência com desenvolvimento de programas para computadores.

Para obter as informações necessárias para a utilização do protótipo, o aluno deve executar o programa e selecionar o menu **Exercícios** e **Exibir** (ver 5.1).

Figura 5.1: Tela de início do protótipo

Selecionando esse menu acima, aparecerá a tela mostrada na figura 5.2 abaixo:

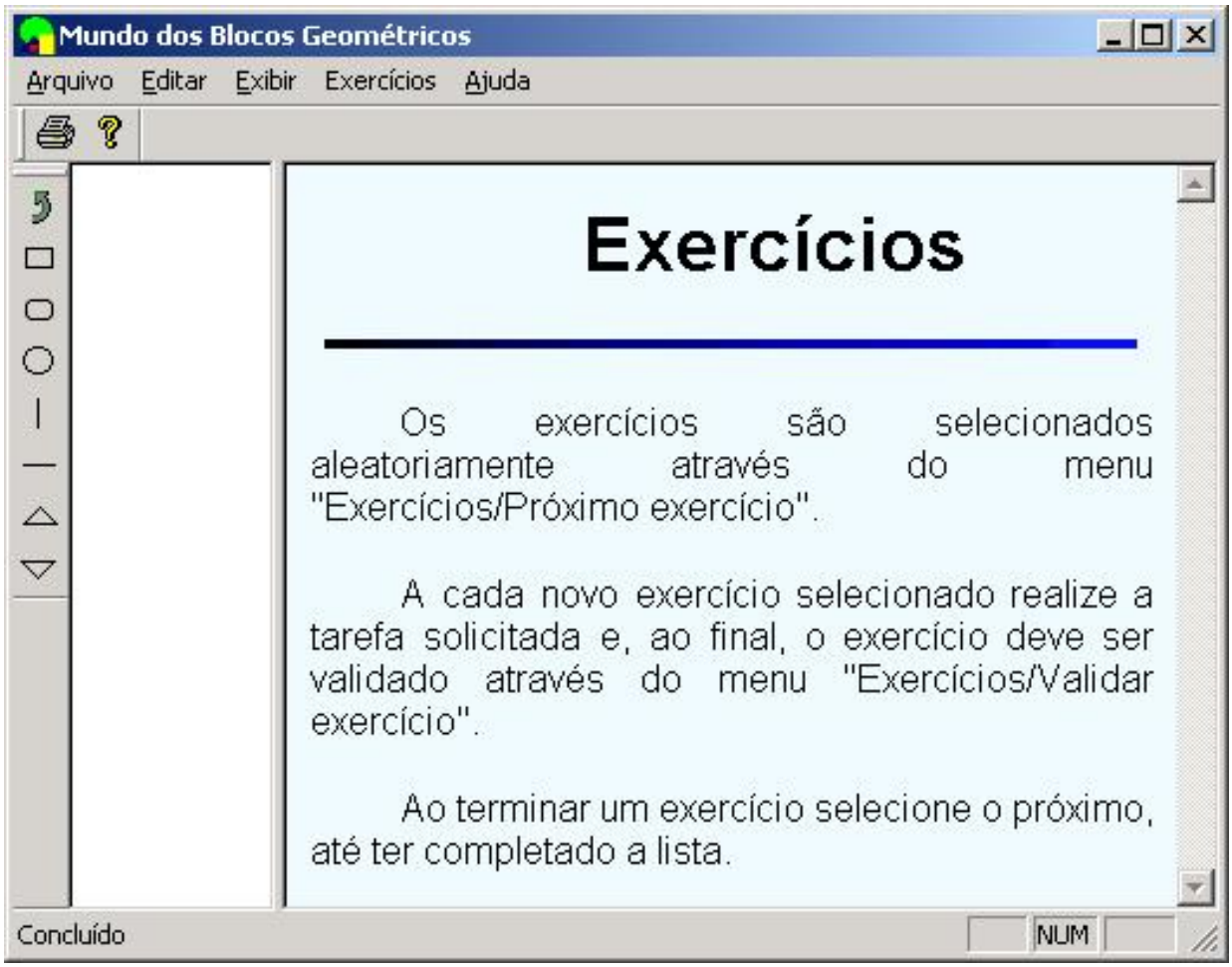

Figura 5.2: Tela de utilização do protótipo

Para iniciar o jogo o aluno, o qual é a parte mais importante do processo, deve pressionar o menu **Exercícios** e **Próximo exercício** onde será, aleatoriamente, sorteado um exercício a ser resolvido pelo aluno.

As instruções para a resolução do exercício, após a escolha aleatória, aparecerão no canto direito da tela.

Visualiza-se isso, através da figura 5.3, que mostra a tela do ambiente, onde o aluno selecionou o exercício aleatoriamente, sendo que o aluno está livre para resolver a atividade, no tempo e quantas vezes achar necessário para o seu aprendizado.

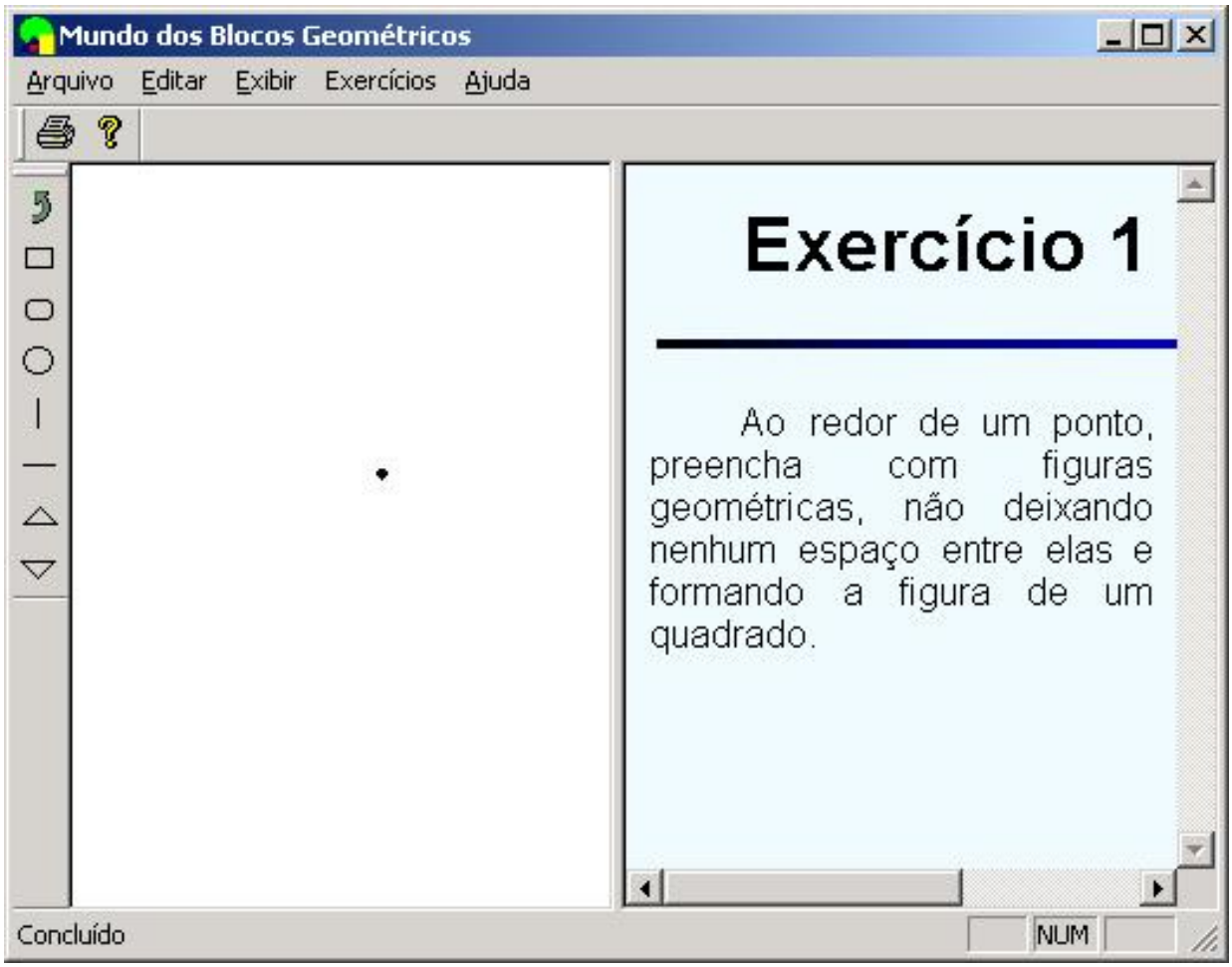

Figura 5.3: Tela com as instruções para o exercício 1

Com o enunciado do exercício escolhido, agora o aluno deve escolher quais figuras geométricas, ele irá utilizar para conseguir chegar à solução do problema proposto.

Para escolher as figuras, deve-se clicar na figura escolhida, que está na barra de ferramentas e clicar com o mouse na tela, no local onde se quer colocar a figura, conforme a figura 5.4.

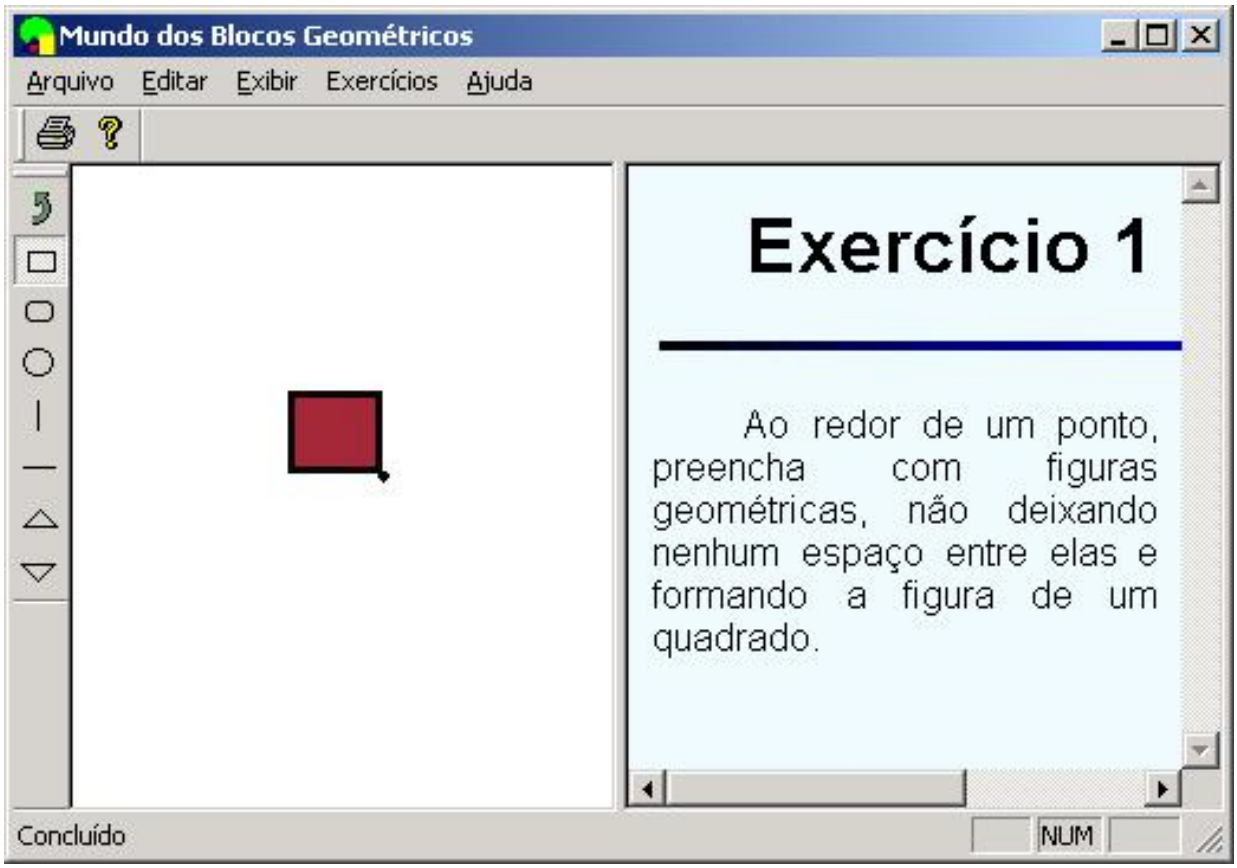

Figura 5.4: Tela com posicionamento das figuras na tela

A realização dos exercícios no mundo dos blocos geométricos está intrinsecamente relacionada com a programação, quando o aluno, via teclado, interage com o computador, escolhendo o exercício que deseja resolver e quais figuras geométricas farão parte dessa solução para montar a figura que atenderá aos requisitos do problema apresentado.

Através da escolha do exercício e da figura geométrica, o aluno estará tendo seus dados de entrada, ou seja, definindo suas funções de entrada, para a solução e com a figura geométrica escolhida ele poderá recursivamente ir clicando na tela e preenchendo o desenho com a figura, aprendendo assim o conceito de recursão.

Enfatiza [14] que atualmente, não só grande parte das linguagens de programação possuem facilidades de recursão, como também a arquitetura de muitos computadores modernos possui estruturas para tratar eficientemente recursão.

O aluno poderá utilizar a figura selecionada quantas vezes achar necessário para atingir o objeto proposto no exercício, demonstramos isso na figura 5.5.

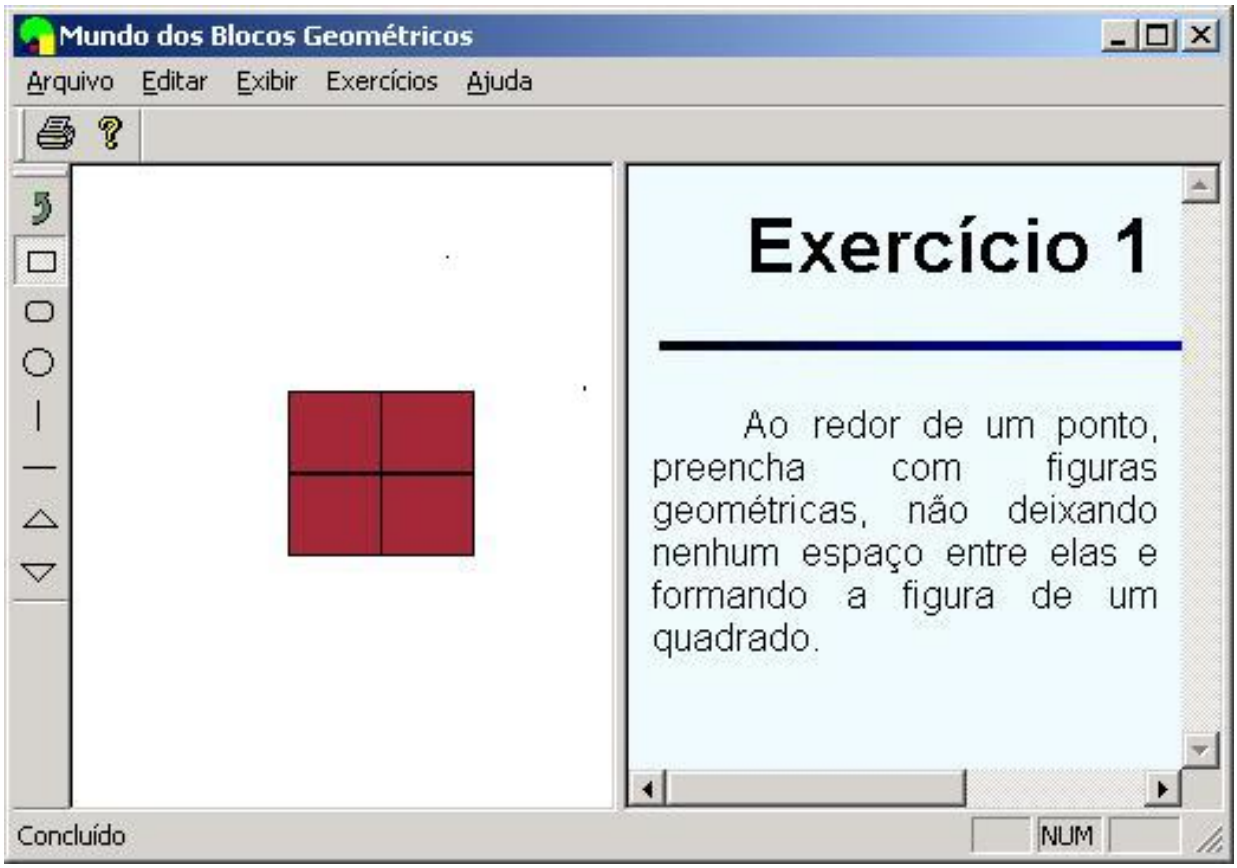

Figura 5.5: Tela com o conceito de recursividade

Para melhor entender-se o conceito de recursão pode-se observar o exemplo das ações de um relógio.

O relógio começa a contar as horas do 0(zero) até 24(vinte e quatro) horas. E se tiver “corda” ou bateria, ele recomeça tudo.

Assim sendo, a recursão faz com que um procedimento seja repetido várias vezes, sendo o que o aluno está praticando no jogo, pois a partir do momento que ele escolhe uma figura geométrica poderá utilizá-la recursivamente, até que seu objetivo esteja alcançado.

Conforme Barreto [5] recursividade é o mecanismo de definir uma função por ela mesma. Por exemplo, uma definição de fatorial de um inteiro é que o fatorial de um número “n” é igual a “n” multiplicado pelo fatorial de “n - 1” , o fatorial de zero sendo 1 por definição.

Se para a resolução do exercício, o aluno necessitar de duas ou mais figuras geométricas, deve selecionar a figura que deseja utilizar, utilizá-la na tela quantas vezes achar necessário e então selecionar outra figura para continuar construindo sua solução do problema e assim sucessivamente para todas as figuras que necessitar.

Com essa interatividade no ambiente proposto, reforça-se o que Brito [9], em seu

artigo declara sobre as palavras de Papert:

Segundo Papert, no construcionismo um aprendiz constrói seu conhecimento mais facilmente, quando está engajado na construção de alguma coisa externa ou, pelo menos, em algo que ele possa dividir com os outros.

Na abordagem construcionista, os projetos se constituem em planos de trabalho e conjunto de atividades que podem tornar o processo de aprendizagem mais dinâmico, significativo e interessante para o aluno, deixando de existir a imposição dos conteúdos de maneira autoritária. Com esse pressuposto, no protótipo a partir da escolha do exercício, o aluno realiza pesquisas, investiga, registra dados, formula hipóteses, analisa, aplica e avalia o artefato construído, para o entendimento de seus conceitos sobre geometria e PF.

De acordo com Santos [51] o processo de aprendizagem deve ser ativo, no exercício operativo da inteligência. O conhecimento não implica em reproduzir o objeto, mas agir sobre ele.

Aprender fazendo, agindo, experimentando é o modo mais natural, intuitivo e fácil de aprender.Isso é mais do que uma estratégia fundamental de aprendizagem: é um modo de ver o ser humano que aprende.

Para concluir o exercício, o aluno deve ir no menu **Exercícios** e **Validar exercício** e se o objetivo do exercício for alcançado, a figura desenhada pelo aluno irá ficar piscando na tela, alertando o aluno que ele conseguiu chegar ao objetivo proposto. Tendo nesse momento ele construído sua função de saída para o problema proposto, conforme a figura 5.6.

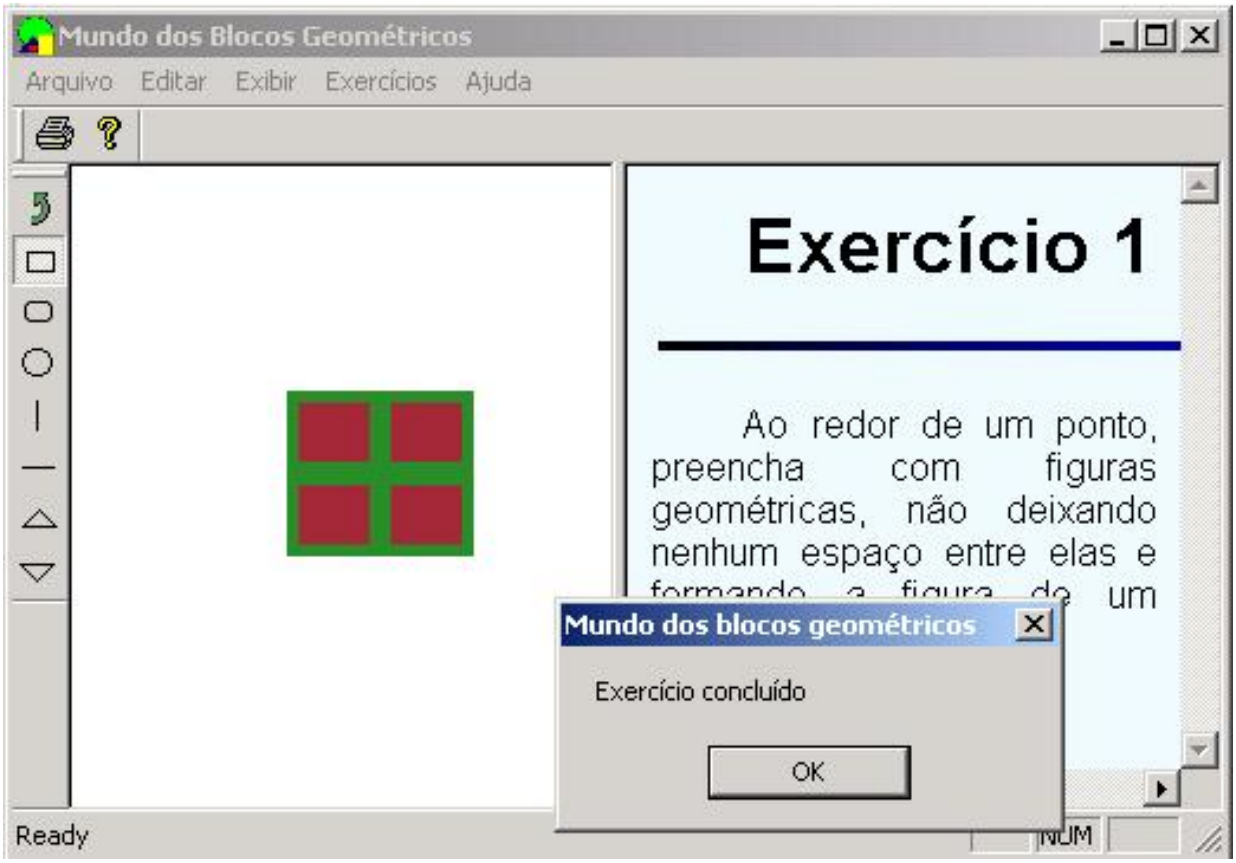

Figura 5.6: Tela com exercício completado corretamente

Se o aluno, clicar no menu **Exercícios** e **Validar exercício** e não tiver atingido o objetivo, o programa enviará a ele uma mensagem alertando-o de que o objetivo não foi alcançado, conforme mostrado na figura 5.7, podendo o aluno refazer a sua análise sobre como desenvolver o conhecimento para atingir o objetivo, ou reavaliar sua ação e modificar sua figura geométrica para conseguir atingir a conclusão correta do exercício, validando-o novamente.

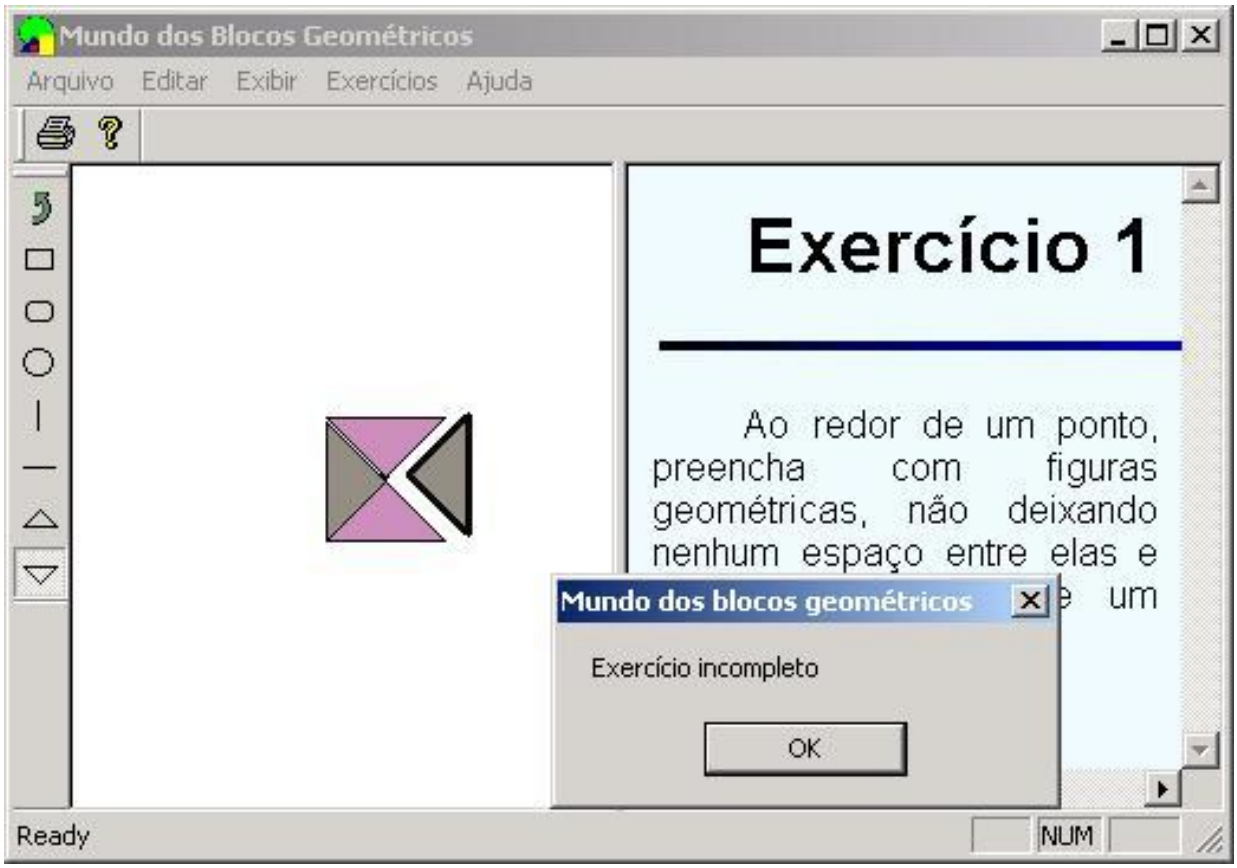

Figura 5.7: Tela com exercício completado incorretamente

Para a solução de alguns exercícios, pode-se ver um deles na figura 5.8 abaixo, teremos que usar o conceito de condição. Que pode ser melhor entendido com o exemplo do exercício: uma determinada figura só poderá fazer parte da solução, se na figura geométrica de duas dimensões que está sendo criada, aparecerem triângulos em branco, entre as figuras utilizadas.

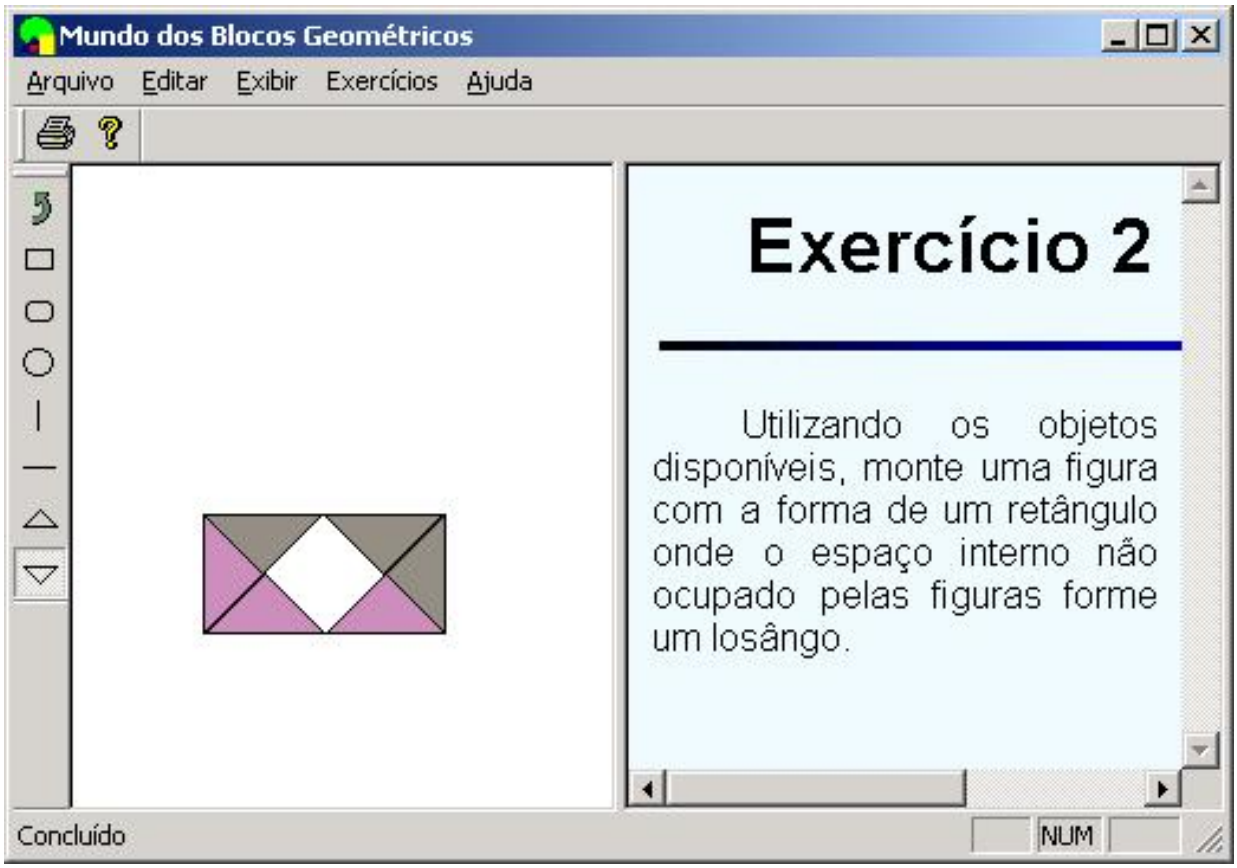

Figura 5.8: Exemplo de resolução do exercício, utilizando condição

Note que são palavras “SE” e “ENTÃO” que determinam a condição.

Pode-se, com um exemplo bem simples entender melhor uma condição, com a utilização da frase: Se não chover, então sua mãe irá ao supermercado.

Observa-se que a condição necessária, para sua mãe ir ao supermercado, é não chover.

A condicional pode nos facilitar a construção de procedimentos complexos, fazendo com que um procedimento faça parte de outro procedimento.

O grande efeito especial acontece quando o aluno, consegue estruturar a situação, na qual sua aprendizagem ocorre, assimilando o processo da sua interação no ambiente com o processo da programação. Aprendendo o conteúdo, ao tentar buscar a solução para um problema, permitindo-se a utilização das mais diversas alternativas e estratégias, de acordo com o seu nível para a resolução dos exercícios no computador, através da escolha das figuras para a solucção do problema (primitivas de entrada), utilização das figuras várias vezes (recursão) e posicionamento das figuras (condição) no local correto para a completa solução do exercício. A compreensão mais profunda dos conteúdos se dá na medida em que o aluno é forçado a refletir sobre os processos envolvidos ao resolver seus exercícios. Desta forma, o aprendizado ocorre necessariamente de acordo com o nível e

o estilo cognitivo do aluno.

Eles aprendem trocando entre si, conscientes de seus próprios processos de aprendizado. [2]

Com a brincadeira de realizar os exercícios, os alunos compreendem a lógica da linguagem do computador através de suas operações mentais trabalhando na máquina virtual, de seus mecanismos de regulação ativa, estabelecendo um tipo de correlação entre as instruções do programa, então como significantes e suas próprias operações lógicas como significados. Os alunos poderiam, nesse nível de conceitualização, antecipar a produção do programa, assim como suas possíveis transformações, a partir de composições e recombinações necessárias.

Segundo Lucena [32], transformar o paradigma tradicional da educação como fábrica para a educação como entretenimento, no sentido de escolha individual da busca da informação necessária ao aprendizado de um determinado assunto, é a questão chave da implantação de novas tecnologias de suporte à educação. Isto porque é preciso que haja interesse e motivação para buscar a informação desejada.

É através dessa busca individual do conhecimento, mas com participação e troca de informações com os colegas e professores, estes últimos que devem ser preparados para trabalhar como facilitadores, tutores e até mesmo provocadores da participação, que se pretende, com as atividades propostas no protótipo, auxiliar os alunos a identificar os pontos principais de determinados temas, a buscar causa e efeito, identificar padrões e relações, ordenar idéias, desenvolver linhas de pensamento, construir taxionomias ou categorizações, fazer comparações e estabelecer contrastes, examinar relações de custos versus benefícios e a interligar idéias.

O objetivo do JOGO é desenvolver o raciocínio do aluno e não transformá-lo em um programador, reduzindo a carga cognitiva inicial, pois, permite trazer a manipulação geométrica para o nível da observação concreta, ou seja, as figuras são objetos concretos que podem ser construídos, destruídos e modificados à vontade. Onde o professor pode usar esse programa como modo de introduzir os conceitos do paradigma funcional, manipulando os problemas apresentados com as figuras geométricas. Pretende-se facilitar a aprendizagem sobre os conceitos da PF, com uma dinâmica de ensino e aprendizagem que estimule os processos cognitivos facilitadores da aprendizagem, fazendo com que os alunos sejam ativos e colaborem na construção do seu saber.

Capítulo 6

Conclusão e sugestões para trabalhos futuros

“Nunca ande pelo caminho traçado, pois ele conduz somente até onde os outros já foram .”(Alexandre Graham Bell)

Uma condição fundamental para a vida nas décadas futuras é desenvolver a capacidade de aprender.

Vive-se um momento da história onde o computador é peça fundamental no desenvolvimento científico e educacional da sociedade.

A introdução do computador na área educacional propõe utilizar a informática como um meio auxiliar para melhorar o processo educacional.

O Mundo dos blocos geométricos exige que o aluno conheça apenas algumas poucas regras básicas para poder começar a usufruir o seu ambiente, sem ter que se preocupar como o computador funciona, utilizando as facilidades da máquina virtual. Contudo, logo se percebe que há muito para aprender.

Pressupõe-se que ele irá motivar, envolvendo o aluno, criando um ambiente apropriado para aprender com prazer. Podendo gerar dinâmicas nas aulas, auxiliando o aluno a se iniciar no pensamento científico.

Aprendendo e observando uma atividade e pensando sobre ela, ordenando esses pensamentos de acordo com suas hipóteses e suas expectativas; compartilhando suas idéias, escutando e valorizando as idéias dos colegas por perceber que elas podem ser úteis para o seu próprio pensamento, enfim auxiliando os alunos a aprender a aprender.

Ao manipular as figuras do programa, o aluno poderá explorar a consistência de suas idéias, de seus modelos mentais, transformando-os em entidades reais, passíveis de análise e, sobretudo de interferências pelos outros alunos ou professores.

O que nos levará à preservação de um dos mais importantes aspectos do uso do protótipo, que é o de possibilitar a reflexão sobre os resultados obtidos e a análise do processo de obtenção dos mesmos. O protótipo resultante dessa dissertação foi elaborado com a intenção de preencher uma lacuna existente na educação de modo diferenciado com relação aos ambientes educacionais lúdicos, que propiciam prazer aos alunos e instigam sua curiosidade e melhoram sua capacidade de aprendizagem.

Fazendo com que o aluno utilize os mecanismos básicos da PF, não por que decorou os comandos e as regras de uso dos mesmos, com a memorização dos conceitos apresentados, mas porque assimilou a mecânica do jogo e conseguiu estimular o desenvolvimento do seu senso crítico e da criatividade, através do uso de sua inteligência e capacidade de raciocínio natural, enfatizando a construção de seu conhecimento. Porque a aprendizagem significativa ocorre quando novos conceitos são relacionados aos conceitos familiares existentes na estrutura cognitiva do aluno e podem ser aplicados a todos os domínios.

Teve-se por objetivo com esse protótipo a intenção de modificar, ou pelo menos, questionar, as práticas de ensino vigentes e também fazer com que o aluno aprenda a aprender de uma maneira prazerosa e natural.

Os computadores estão cada vez mais importantes em nossa sociedade e a escola precisa fazer com que essa “máquina tão poderosa” e necessária faça parte do conteúdo das disciplinas.

Utilizando o computador não como um meio de transferir a informação, mas sim como um meio de construir o conhecimento.

O desafio em mudar a forma de ensinar e aprender no contexto da escola reside em criar ambientes de aprendizagem que incentivem o uso de diferentes ferramentas para enriquecer a exploração e a investigação de um problema para dar origem a outros problemas.

Verificando erros e acertos, observando as reações dos alunos, suas preferências, ouvindo opiniões, críticas e sugestões no dia a dia.

Acredita-se que a utilização do protótipo apresenta alguns aspectos relevantes que podem significar um salto qualitativo na educação: capacidade para individualizar a aprendizagem, trabalho cooperativo, desenvolvimento do espírito crítico, novas perspectivas para o trabalho do professor.

6.1 Sugestão para trabalhos futuros

- Avaliação do protótipo mediante sua utilização em uma escola com o acompanhamento de professores;
- Implementação de mais níveis de exercícios com a possibilidade de colocar comandos, através de uma linguagem simples, com comandos funcionais prontos que gerem funções para solução de problemas matemáticos simples, fazendo com que o aluno, que já conseguiu assimilar os mecanismos básicos de PF através da geometria possa encontrar nos próximos níveis, mais motivação e novos desafios para continuar a construção do seu saber e desenvolvimento do seu raciocínio;
- Habilitar o protótipo para uso na internet, permitindo que os alunos tenham a oportunidade de interagir com o programa de onde quiserem, facilitando a comunicação e a colaboração para o aumento de seu aprendizado individualizado.

Não se esgotou, no entanto, os recursos desta nova maneira de estimular a vontade de aprender, presume-se que o protótipo trará ganhos reais e maiores para a educação à medida que forem feitas experiências com os alunos em sala de aula e forem feitas implementações mediante o desenvolvimento cognitivo dos alunos.

Para se ter certeza que o computador esta ajudando a transformação do ensino, entrando no sistema educacional para alimentar o processo de aprendizagem natural e espontâneo dos alunos.

Através de diversas atividades, os alunos são levados a perceber que o uso dos recursos informatizados não é um fim em si mesmo, mas um meio para desenvolvimento, construção, expressão e integração do conhecimento. Isso significa que o importante não é o tipo ou marca de determinado recurso, e sim o uso que possa ser feito dele a nível pessoal ou profissional.

Referências Bibliográficas

[1] ABREU, R. A. S. Uma avaliação sobre o uso da linguagem logo no processo de construção de noções topológicas. Dissertação de Mestrado, Pontifica Universidade Católica, Rio de Janeiro, 1990.

[2] ALMEIDA, F. J. E FONSECA JÚNIOR, F. M. *Proinfo: Projetos e Ambientes Inovadores*. Ministério da Educaccão , Seed, Brasília, 2000.

[3] ALMEIDA, M. *Hipertômatos na Computação Aplicada à Educação*. Tese de Doutorado, Universidade Federal de Santa Catarina, Departamento de Ciência da Computação, Florianópolis, fevereiro 2002.

[4] ALONSO, C. M. M. C. & MEDINA, R. D. Ambientes informáticos baseados em realidade virtual como instrumento de aprendizagem para pessoas portadoras de deficiências. *International Conference on Engineering and Computer Education* (2003).

[5] BARRETO, J. M. *Inteligência Artificial no Limiar do Século XXI*, 3 ed. $\rho\rho\rho$ Edições, Florianópolis, SC, 2001.

[6] BARROS, C. S. G. *Pontos de Psicologia Escolar*. Editora Ática, São Paulo, 1989.

[7] BIRD, R.; WADLER, P. *Introduction to Functional Programming*. Hertfordshire: Prentice Hall, 1988.

[8] BOSSUET, G. *O Computador na escola: sistema LOGO*. Artes Médicas, Porto Alegre, 1985.

[9] BRITO, S. R.; TAVARES, O. D. L. & MENEZES, C. S. Mediador - um ambiente para aprendizagem orientada a projetos com suporte inteligente à mediação. In *Anais do XIII Simpósio Brasileiro de Informática na Educação*, São Leopoldo, novembro de 2002, p. 116–124.

[10] CANIATO, R. *Com Ciência na Educação: Ideíario e Prática de uma alternativa Brasileira para o ensino da Ciência*. Papirus, São Paulo, 1987.

[11] CASTRO, T. H. C. D. E. A. Utilizando programação funcional em disciplinas introdutórias de computação. *Trabalho apresentado no X Workshop sobre Educação em Computação/WEI(Seção Técnica II B) Anais do XXII Congresso da Sociedade Brasileira de Computação* (2002), 157–168.

[12] CHAIBEN, H. Um ambiente computacional de aprendizagem baseado em redes semânticas. Dissertação de Mestrado, Centro Federal de Educação Tecnológica do Paraná, Departamento de Informática Industrial, Curitiba, 1996.

[13] CHAVES, E. & SETZER, V. *O Uso de Computadores em Escolas: Fundamentos e Críticas*. Scipione, São Paulo, 1987.

[14] DIVERIO, T. A. *Teoria da Computação: máquinas universais e computabilidade*. Editora Sagra Luzzatto, Porto Alegre, RS, 2003.

[15] FAGUNDES, L. *Psicogênese das Condutas Cognitivas da Criança em Interação com o Mundo do Computador*. Tese de Doutorado, Instituto de Psicologia, Universidade de São Paulo, São Paulo, 1985.

[16] FAGUNDES, L. E MOSCA, P. R. A interação criança x computador. *SUCESU São Paulo* (1983).

[17] FAGUNDES, L. E MOSCA, P. R. Interação com computador de crianças com dificuldades de aprendizagem: uma abordagem piagetiana. *Arquivos Brasileiros de Psicologia Rio de Janeiro* (1985).

[18] FAGUNDES, L. E MOSCA, P. R. As conceitualizações das crianças que estão programando em logo: a construção e a composição de módulos na imagem mental e na programação. *Arquivos Brasileiros de Psicologia Rio de Janeiro* (1986).

[19] FERNANDES, J. H. C. Ensino introdutório de computação e programação: Uma abordagem concreta, construtivista, lingüística e histórica. *Trabalho apresentado no X Workshop sobre Educação em Computação/WEI(Seção Técnica II B) Anais do XXII Congresso da Sociedade Brasileira de Computação* (2002), 139–146.

[20] FRANCO, V. E. A. I. Na magia de um teclado, os aprendizes do futuro. *Revista INFO Julho*, Número 30 (1985).

[21] FRIEDMAN, D. & FELLEISEN, M. *The Little Lisper*. Trade Edition., The Mit Press, Cambridge, Massachusetts and London, England, 1987.

[22] GASPERETTI, M. *Computador na educação: guia para o ensino com as novas tecnologias*. Editora Esfera, São Paulo, 2001.

[23] GOODYEAR, P. *Logo: introdução ao poder do ensino através da programação*. Campus., Rio de Janeiro, 1986.

[24] GOUVEA, R. *Recreação.* Rio de Janeiro, 1967.

[25] HASEMER, T. & DOMINGUE, J. *Common Lisp programming for Artificial intelligence* . Mackays of Chatham plc, Chatlham, Kent., Great Britain, 1989.

[26] HUIZINGA, J. *Homo ludens.* Perspectiva, São Paulo, 1999.

[27] KREUTZ, L. S. Modelo computacional para ensino de fisiologia cardiovascular. Dissertação de Mestrado, UFSC, Departamento de Informática e de Estatística, Florianópolis, Fevereiro 2001.

[28] LA TAILLE, Y. D. *Ensaio sobre o lugar do computador na educação*. Iglu Editora, 1990.

[29] LIEBERMAN, N. *Playfulness: its relationship to imagination and creativity*. Academic Press, New York, 1977.

[30] LINS, R. D. *Considerações sobre a implementação de linguagens funcionais.* UFRJ/NCE, Rio de Janeiro, 1988.

[31] LOLLINI, P. *Didática e computador - Quando e como a informática na escola*. Edições Loyola, São Paulo, 1985.

[32] LUCENA, C. E FUKS, H. *Professores e aprendize na Web: a edugção na era da Internet.* Clube do Futuro, Rio de Janeiro, 2000.

[33] MARTINS, A. & FERNEDA, E. Aprendizagem por analogia na máquina e nas pessoas. *Revista de Informática Teórica e Aplicada III*, 1 (1996).

[34] MAURER, W. D. *The Programmer's Introduction to Lisp.* Hazell Watson Viney Ltd. , Aylesbury, Bucks, Great Britain, 1972.

[35] MEIRA, L. R. L. Geometrias em ação na programação em logo. Dissertação de Mestrado, Unicamp, Núcleo de informática aplicada à educação, Recife, 1987.

[36] MEIRA, S. R. L. *Introdução à Programação Funcional* . Unicamp, IMECC, Campinas, 1988.

[37] MEIRIEU, P. *Aprender... sim, mas como?* Artes Médicas Sul, Porto Alegre, 1996.

[38] MENDONÇA, F. D. V. S. D. *LOGO: projetos.* McGraw-Hill, São Paulo, 1989.

[39] MORAES, R. D. A. *Informática na Educação.* DPA, Rio de Janeiro, 2000.

[40] NOVO, I. *Biblioteca Básica: Informática Lisp 30 uma nova linguagem.* Século Futuro, Rio de Janeiro, 1986.

[41] ONETTA, A. A. O problema do ensino dos números inteiros dentro da matemática e a apresentação de um protótipo alternativo valorizando o uso dos jogos. Dissertação de Mestrado, UFSC, Departamento de Ciências da Computação, Florianópolis, 2002.

[42] PAGANO, R. *Computer simulation as an educational tool*. Tese de Doutorado, Faculty of Applied Sciences, University of Louvain, Louvain la Neuve, Bélgica, dezembro 1992.

[43] PAPERT, S. *Mindstorm: Children, Computers and Powerful Ideas*. Basic Books Inc, New York, 1980.

[44] PAPERT, S. *Logo: Computadores e Educação*. Editora Brasiliense, São Paulo, 1985.

[45] PERRENOUD, P. *Construir as Competências Desde a Escola*. Artes Médicas Sul, Porto Alegre, 1997.

[46] PRADO, B. B. E. E. A. *Parâmetros, Condicionais, Recursão...Continuando o passeio pelo Logo.* NIED-UNICAMP, Campinas, 1997.

[47] RABUSKE, R. A. *Inteligência Artificial.* Ed. Da UFSC, Florianópolis, 1995.

[48] ROCHA, H. V. O uso da linguagem logo em curso de introdução à programação de computadores para alunos ingressantes no bacharelado de ciência de computação. *Publicação do Núcleo de Informática Aplicada à Educação da Unicamp/NIED: Campinas/SP Memo* N^0 *10* (1988).

[49] RODRIGUES, M. C. R. J. Como ensinar programação? *Computação Brasil publicado pela Sociedade Brasileira de Computação/SBC Ano III*, Edição 7 (2002), 05.

[50] SANTAROSA, L. M. C.; BARBIERI, E. G.; MACHADO, R. K. & PETERSON FILHO, R. A. *Manual LOGO.* Editora da Universidade/ UFRGS; MEC/ SESu / PROEDI, Porto Alegre, 1988.

[51] SANTOS, M. A. P. A construção do número e das figuras geométricas na programação logo. Dissertação de Mestrado, Pontificia Universidade Católica do Rio de Janeiro, Rio de Janeiro, 1990.

[52] SCHMITT, F. Protótipo de um ambiente para o ensino de algoritmos. Relatório técnico., Curso de Pós-Graduação em Nível de Especialização em Tecnologias de Desenvolvimento de Sistemas - Universidade Regional de Blumenau - FURB (Orientador: C. Odebrecht), Blumenau, 1998.

[53] SEBESTA, R. W. *Conceitos de linguagens de programação.* Bookman, Porto Alegre, 2000.

[54] SILVA, L. S. DA E GARCIA, L. F. F. Gutemberg - ferramenta de hipermeios para aplicações educacionais. *Simpósio Brasileiro de Informática na Educação - Porto Alegre*, Edição 7 (1994), 152–163.

[55] THOMPSON, S. *Haskell, The Craft of Functional Programming.* Essex: Addison-Wesley, 1996.

[56] TJARA, S. F. *Informática na Educação: professor na atualidade.* Érica, São Paulo, 1998.

[57] URRUTH, B. J. D. Análise de alguns softwares gratuitos relacionados com a matemática do segundo grau. Dissertação de Mestrado, UFSC, Departamento de Ciências da Computação, 2002.

[58] VALENTE, J. A. *Logo: Conceitos, aplicações e projetos.* Mc Graw-Hill, São Paulo, 1988.

[59] VALENTE, J. A. *Por Quê o Computador na Educação.* Gráfica da UNICAMP, São Paulo, 1993.

[60] WYGOTSKY, L. S. *A Formação Social da Mente: o desenvolvimento dos processos psicológicos superiores.* Editora Martins Fontes, São Paulo, 1984.